M/M/1 POLLING MODELS WITH TWO FINITE QUEUES

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Abdullah Dayar

September, 1995

M/M/1 POLLING MODELS WITH TWO FINITE QUEUES

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL

ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Abdullah Daşcı

September, 1995

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. M. Cemal Dinçer (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. M. Selim Aktürk

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Halim Doğrusöz

Approved for the Institute of Engineering and Sciences:

Prof. Mehmet Baray
Director of Institute of Engineering and Science

ABSTRACT

M/M/1 POLLING MODELS WITH TWO FINITE QUEUES

Abdullah Daşcı
M.S. in Industrial Engineering
Supervisor: Assoc. Prof. M. Cemal Dinçer
September, 1995

Polling models are special kinds of queueing models where multiple-customer type single-stage is considered. In this thesis, first an overview and a classification of polling models will be given. Then two-costomer one server M/M/1 polling models will be analyzed and the performance of models will be developed for exhaustive, gated, and G-limited service policies. We give analytical methods for a special type of polling model where we solve the system to get mean queue lengths and thruput rates by three methods. The first one is based on solving the steady state distribution of the Markov Process. The second is a decompositon aiming to decrease the size of the problem. The third one is an approximation method that uses the earlier results and it is very accurate. The thesis will be concluded with possible future extensions.

Key words: Markov Processes, Queueing Theory, Regenerative Processes, Polling Models, Performance Evaluation, Classification.

ÖZET

İKİ SONLU KUYRUKLU M/M/1 SEÇMELİ KUYRUK MODELLERİ

Abdullah Daşcı
Endüstri Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Doç. M. Cemal Dinçer
Eylül, 1995

Seçmeli kuyruk modelleri çok müşterili tek işgörenli olarak kuyruk modellerinin özel bir halidir. Bu tezde önce seçmeli kuyruk modellerinin bir sınıflandırması yapılacak ve bu modellere genel bir bakış verilecektir. Daha sonra özel bir tek işgörenli seçmeli kuyruk modeli üzerine, tümden, köprülü ve G-kısıtlı servis politikalarında ortalama çıktı hızı ve ortalama kuyruk uzunluğu üzerine çözümsel yöntemler verilecektir. Birinci yöntem bir Markov sürecinin çözülmesidir. İkinci yöntem problemin büyüklüğünü azaltmaya yönelik bir ayrıştırmadır. Üçüncü yöntem ise daha önceki sonuçları kullanan oldukça hassas bir yaklaştırmadır. Tez ilerisi için öngörülen çalışma konularıyla sonuçlandırılacaktır.

Anahtar sözcükler: Markov Süreçleri, Kuyruk Kuramı, Yeniden Üremeli Süreçler, Seçmeli Kuyruklar, Performans Değerlendirmesi, Sınıflandırma.

To my parents

ACKNOWLEDGEMENT

I am indebted to Assist. Prof. M. Cemal Dinçer for his invaluable guidance, encouragement and above all, for the enthusiasm which he inspired on me during this study.

I am also indebted to Prof. Dr. M. Halim Doğrusöz and Assist. Prof. Selim Aktürk for showing keen interest to the subject matter and accepting to read and review this thesis.

I would like to thank to Dr. Nureddin Kırkavak for his invaluable comments and help during the garaduate life.

I would also like to thank to my classmates Engin Topaloğlu, Alper Şen, Abdullah Çömlekci, Yavuz Karapınar, Mustafa Karakul, Erdem Gündüz and Selçuk Avcı for their friendship and patience.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Queuing networks have been widely used to estimate the various performance measures of production systems after 1950s. As Leung and Suri [21] points, they are most appropriate for aggregate analysis in the design phase to explore a large number of alternative system designs, and for production and capacity planning in the operational stage. Most of the systems can be modeled in a form of queuing network. But huge computational and memory requirements prevent researchers to analyze the real systems accurately. Most real life systems have various control policies, large number of machines, multiple type products, general processing, failure and repair time distributions, hence they are very difficult to model and solve as a queuing system. Thus, most researchers deals with single-item multiple-stage, or multiple-item single-stage models, in which some of the distributions are Markovian.

In this thesis, a two customer, single server polling system is considered that has been studied extensively to model both production systems, computer networks, and satellite communication systems. For example, in computer local area networks (LANs), the nodes represent terminals or workstations. The customers are the messages that arrive at a terminal for processing. The server is an imaginary "token" or signal that activates a terminal whose messages are then transmitted or processed. After a delay for switching, the token then goes to the next node and activates it. This represents the "token ring" type

of LAN protocol [28]. Another application is the stochastic multi-product production systems where the server is a machine or cell of machines and the nodes correspond to different product types by considering customers as the orders. Using polling models some performance measures can be calculated such as, thruput rate, average queue length, mean waiting time and so forth.

In polling models there are different customers which have individual arrival processes and service times. Arrivals join their dedicated queues. A server visits these queues in an order, which may be predetermined as well as random. At each visit server gives service according to a given policy. Customers for which the service is completed departs from the system. A switchover time elapses as the server moves from one queue to another.

In this thesis queuing theory terminology will be used. Products are referred as customers, as well as machine or processor as server, and setups as switchovers.

Organization of the thesis is as follows: In the next chapter a classification of one server models and a literature review will be given. The third chapter is devoted to analyze of M/M/1 polling models under three service policies. Numerical results of the different service policies including the algorithms are provided in Chapter 4. The thesis ends with the conclusion and suggested avenues for future research.

Chapter 2

Polling Models: A Classification

There have been many alternative forms of polling models as seen in the literature. The common characteristics among all models are the existence of randomness. Also all works deal with stationary distributions. Most of them assume that there is only one server, although polling models find applications with multiple servers. This classification based on the articles that have been published since 1966. Although it is not claimed to be an exhaustive survey it will be very helpful in analyzing the previous works, assessing their contribution in the literature. First of all, the differentiating attributes of polling models will be analyzed. These are arrival process, service time, switchover time, polling policy, service policy, service discipline, buffer size, customer types, preemption, symmetricity, vacation and unreliability.

1. Arrival Process is the first distinguishing element of polling models. Most of the literature deals with Poisson arrivals, but there is a work ([38]) that consider renewal processes. So interarrival times could be :

i. Deterministic. or

ii. Stochastic

- . Exponential,
- . Phase type,
- . General-continuous,
- . General-discrete.

Nevertheless there is no model that has deterministic, or continuous and generally distributed interarrival time in any work.

2. Service time has almost similar characteristics with the arrival process. Although most of the literature deals with general distributions, there are some work that consider Markovian distributions. So service times could be:

i. Deterministic, or

ii. Stochastic

- . Exponential,
- . Phase type,
- . General-continuous,
- . General-discrete.

3. Switchover Time or setup time is a complicating element in the polling models. When server moves from one queue to another it is idle for a period of time. This period includes all the elements of preparing for another queue. Switchover times could be :

i. Deterministic

- . Zero,
- . Non-zero.

ii. Stochastic

. Exponential,

. Phase type,

. General-continuous,

. General-discrete.

There is a considerable amount of studies ([27],[14],[8]) that consider zero switchover time. Solution techniques may be different whether swithcover times are

. Queue dependent, or

. Sequence dependent.

4. Service policy may be the most differentiating feature of polling models. It defines either probabilistically or deterministically the number of customers to be served by the server in a visit.

These policies can mainly be divided into two parts: dynamic policies and static policies. Dynamic policies are mostly encountered in the optimization models, where service policy is determined by the system state. But service policies are static in most performance evaluation models. In the static case the policy does not change in time, whatever the system state is. These static models are explained in detail.

There are mainly three static service disciplines that have been studied extensively in the literature: exhaustive, gated and limited. Once server begins to serve a queue, it continues until queue becomes empty, this is known as exhaustive service policy. Gated service policy is the same as exhaustive, but arrived jobs after server's visit time are left to the next visit. Limited case refers to a fixed batch. Server always processes up to a fixed amount from a particular queue, then switches to the next queue. But there appears different policies in case of server faces with less than batch size of customers. They are explained later in this section. When the batch size is one that policy is called

as 1-limited or non-exhaustive. Besides, there are less popular policies which are the generalizations of the first three. Binomial-gated policy is an extension to gated one. In this policy, the number of customers served by the server is binomially distributed with parameters of n, number of customers present in the visit instant, and p, a fixed queue dependent probability. When $p = 1$ this policy reduces to gated one.

E-limited is a hybrid of limited and exhaustive, server decides to serve queue exhaustively or a randomly chosen number of customers are served. An example of E-limited policy is Bernoulli. After a customer is served, server decides to give service to another customer with a queue dependent probability, p, otherwise visit finishes. When $p = 1$ and $p = 0$ this policy reduces to exhaustive and to 1-limited respectively. G-limited is also a special case of limited service policy. Server serves a fixed number of customers when it finds more customers than this limit at the polling instant, otherwise he serves all the customer present at that instant and leaves the new arrivals to the next visit.

There also exist some other service policies reported in the literature. In semi-exhaustive policy server continues serving until the number of customers drops to one less of the number at the polling instant. For instance server leaves the queue if there is no new arrival until the service completion of the first customer, otherwise he continues with the second one, and so forth. Time limited policy refers to give service in time units rather than number of customers. The distributional assumptions about the time limits adds more classes to polling models.

As a result, service policies could be summarized as follows:

a. Dynamic policies, or

b. Static policies

   i. Bernoulli

      . $p = 1$ (Exhaustive),

. $0 < p < 1$,

. $p = 0$ (1-limited or non-exhaustive).

ii. Binomial gated

. $p = 1$ (Gated),

. $0 < p < 1$.

iii. Limited

. $K = 1$ (1-limited or non-exhaustive),

. $1 < K < \infty$ (K-limited),

. G-limited,

. E-limited.

iv. Semi-exhaustive

v. Time limited

. Constant,

. Exponential,

. General.

5. Polling Policy determines the next queue to be visited. First of all we can classify polling policy as dynamic and static. Dynamic policies are mostly used in optimization models. Here next queue to be visited is determined by the system state. In static policies polling may be determined either probabilistically or deterministically. In the latter case server strictly follows the polling table, in the former case the next queue is selected probabilistically. There are famous deterministic polling policies that extensively studied in the literature. They are cyclical $(1, 2, \ldots, N, 1, 2, \ldots)$, scan type $(1, \ldots, N-1, N, N-1, \ldots, 1, \ldots)$, and star type $(1, 2, 1, 3, \ldots, 1, N, 1, 2, 1, \ldots)$. There can be other policies that are predetermined, fixed, and need not be in a special form. So with respect to polling policies considered, polling models can be classified as:

a. Dynamic,

b. Static.

i. Nondeterministic, or

ii. Deterministic

. Cyclical,

. Scan type,

. Star type (or priority),

. General.

6. Service discipline within queues adds one more attribute as in classical queuing models. It simply determines the order of customers that will be served. Service discipline in a polling model could be:

. FIFO,

. LIFO,

. RO (random order),

. processing time dependent (SPT, LPT, etc.).

7. Buffer sizes are another characteristic of polling models. Solution techniques largely differ when queues have infinite, finite or single capacities. Takagi [35] points out the importance of queue capacities, therefore, queue capacities could be:

. single,

. finite,

. infinite.

8. Number of customer types is also considered as another classification item. Earlier works mostly deal with two customer types. But applications of

polling models have been extended to multiple customer types. But a classification due to the number of customers is inevitable because it is not possible to adopt every solution technique for two customers to arbitrary number of customers. Hence, the number of customer types could be:

. two, or

. multiple.

9. Preemption is also an important modeling feature in polling models as it has been in scheduling theory. Preemption is closely related with the service policy. For static service policy models talking about preemption is meaningless except for time limited case. In a time limited model the question of "what will be done" arises if the time expires in the middle of processing. Also in dynamic service policies the analyst faces with a similar question. Therefore, in a polling model preemption could be:

i. Not allowed, or

ii. Allowed

  . preempt-resume,

  . preempt-repeat,

If preemption is not allowed, server completes his current work and then he proceeds regularly. If the preemption is allowed, there are two cases: If system is preempt-resume, server leaves his work and when returned to the same customer he continues his service for the remaining work. In preempt-repeat case server begins service from the beginning. If the service time has a stationary memoryless distribution these two cases are probabilistically equivalent.

10. Symmetric vs. nonsymmetric distinction is very important in analyzing polling models. When the customers have the same characteristics the system is called symmetric. Symmetric systems are obviously easy to

analyze. There are some works that express the weighted average of waiting times of customers as system parameters. Such representations is called *pseudo-conservation laws.* When the system is symmetric one can easily calculate the various performance measures of the system in closed forms. Hence, the system could be:

. symmetric, or

. nonsymmetric.

11. Vacation is another complication that is added to the polling models. When the system is empty, there are no customers waiting or being served in any queue, a random vacation is scheduled for the server. The distributional assumptions on vacation periods add more classes to the polling models. But there is little work that assume vacation in the model. So, the system could be:

i. Without vacation, or

ii. With vacation

. exponential,

. phase-type,

. general-continuous,

. general-discrete.

12. Unreliability of the server adds another complication to the polling models. Although most of the literature assumes no failure of the server, unreliable servers may be suitable to model production systems where some machines are failure prone. Also, the assumptions about failure and repair time distributions can add more classes. In a polling model, server could be:

i. Reliable, or

ii. Unreliable, and associated distributions could be

- exponential,
- phase-type,
- general-continuous,
- general-discrete.

After this classification it will be easier to understand the real assessment of earlier works and our models.

2.1 Literature Review

Although there are several classes of polling models, studies in the literature are not so evenly distributed among various classes. Instead, the most of the polling models assume that arrivals are Poisson, service and switchover times are generally distributed. Furthermore queues have no limit, they have infinite capacity, number of the queues are more than two. Also service discipline is FIFO. Usually all models fulfills these assumptins. It will be pointed if there is a deviation from these assumptions. While reviewing the literature it is beneficial to divide them into two categories. Infinite queue polling models are reviewed first.

2.1.1 Infinite Queue Polling Models

The earliest work, we are aware of, is due to Skinner [30]. He worked on a two queue system where switchover times consist of a variable and a constant portion. He classifies queues as low priority and high priority. His policy is simply serving high priority queue exhaustively, and low priority queue as 1-limited. He first analyzes an $M/G/1$ queue with vacations and adopts his results to the two queues system. He found the exact Laplace-Stieltjes transform of the waiting time distribution and probability generating function of the queue length

distribution. He also states the stability conditions. Durr [10] analyzes a two-queue system, where one queue has priority over the other. That is whenever high priority customer exists in the system it is served immediately. He considers exponential service and negligible switchover times. He worked on both preemptive and nonpreemptive case and found that:

$$Var(W_i : FIFO) < Var(W_i : RO) < Var(W_i : LIFO)$$

where W_i is waiting time of the ith customer type. Sykes [33] is among the first that include nonzero switchover time. Mean waiting, mean queue length and average busy period are found for exhaustive service in both queues.

Eisenberg [11] works on two queues polling system under alternating priority and strict priority. Alternating priority is simply exhaustive case. In strict priority, the customer type is served immediately as it arrives. Preemptive case is also considered. He obtains mean waiting time and mean queue length. In a later work [12] he finds Laplace-Stieltjes transform of the waiting time distribution, with more than two queues under exhaustive service discipline. He also provides a method to calculate the means of the performance measures. It is the first study analyzing more than two queues

More recent works usually consider arbitrary number of queues and more general polling and service policies. Manfield [23] uses a polling model to find the mean waiting time of messages that arrives to and departs from a central device. Departing messages are generated by the processing of incoming messages, and he argues that outgoing messages are also Poisson and conjectures that outgoing messages should have high priority over the incoming ones in order to refrain from deadlocks. A star type service discipline is assumed. He polls incoming messages non-exhaustively, and outgoing messages non-exhaustively or exhaustively. He finds the Laplace-Stieltjes transform of the waiting time distribution and argues that analysis of outgoing messages is exact, but others are approximate. Boxma et al. [5] provide a pseudoconservation law for cyclical polling models where different service policies applied for queues. In the literature pseudoconservation law is defined as an expression for the weighted average of mean waiting times. They are particularly

important to find performance measures of symmetric systems and checking the correctness of the analytical solutions. Boxma et al. [5] assume discrete time queues where interarrival times, service times and switchover times are generally distributed. They also convert their law to continuous case by applying a limiting procedure. They applied their law to the star polling policy and agree with Manfield [23]. Giannakouros and Laloux [15] generalizes the work of Manfield [23] using the results of Boxma et al. [5]. Constant switchover time is assumed. They model the system when low priority queues are polled not only non-exhaustively but also exhaustively and gated. They showed that non-exhaustive case is very accurate when the traffic intensities are low or medium, and exhaustive and gated cases are very accurate for the whole class of traffic intensities. They also report some numerical results. Takagi and Murata [36] give an exact analysis of waiting time in scan type policy. In their models, service times are constant and interarrival and switchover times have general discrete distributions. They considered three service disciplines, time limited, in which switchover times are assumed to be zero, gated and exhaustive. They also compared these policies with respect to their mean waiting times. Levy [22] finds a pseudoconservation law for cyclical polling system where service policy is binomial-gated. He also finds the control variables, p_i's so as to minimize the sum of weighted waiting times. Watson [40] gives conservation laws for exhaustive, gated, and and non-exhaustive service disciplines. He also gives a brief survey of the work on polling models.

Sarkar and Zangwill [28] calculate mean and variance of cycle time more effectively than previous works. They also note some applications of polling models. In further work [29] they study the effects of processing time and switchover time variation on the system performance. They conjecture that variability play a central role in effective capacity through some paradoxical examples.

Takagi [34] gives a unifying work that considers general priorities within queues in a cyclic polling system. These priorities are FCFS, RO, LCFS, shortest processing time (SPT). That work also studies preempt-resume and nonpreemptive systems.

Ibe and Trivedi [17] study a two-queue case where the server is subject to breakdowns. Failure times are exponential and repair times are generally distributed. They obtain approximations for mean waiting times for 1-limited service policy.

Leung [20] studies on an exponential time limited polling model. He construct an embedded Markov Chain at the visit beginning, visit completion, service beginning and service completions epochs. Waiting time and queue length distributions are found using discrete Fourier Transform. Exponential time limited policy is compared with constant time limited and k-limited policies.

Aforementioned works consider static service policies. But considerable works are studied for optimization models through dynamic service policies. Blondia [4] studies a polling system where polling is made dynamically. Queues are assigned some priority, and after each service completion next queue selected according to this priority. System is nonpreemptive. When the system becomes empty server takes a vacation that is generally distributed. Steady state queue length distribution and Laplace transform of the waiting times are obtained.

Cohen [8] studies a two queues system, where switchover times are assumed to be zero. Service policy is a dynamic one, after each service completion, server visits the queue that has the largest number of customers waiting for service. Ties are broken randomly with predetermined probabilities.

Boxma et al. [6] construct a model to find the optimal visit frequencies of queues in cycle, so as to minimize the sum of weighted waiting times. Since the optimization problem is very hard to solve they propose an approximate method. Also they propose a way to distribute these frequencies evenly in a cycle. They give extensive numerical results to investigate how far away their method from the optimal solution for service policies exhaustive, gated and 1-limited.

Srinivasan [31] works on a nondeterministic polling system. Server, following service at station i, either polls station j with probability p_{ij} if there is a service at station i, or polls station j with probability e_{ij} if there is no service at station i. Obviously $\sum_j p_{ij} = 1$ for all i and $\sum_j e_{ij} = 1$ for all i. He works on exhaustive, non-exhaustive, gated and semi-exhaustive service policies. He obtains expected cycle time and stability conditions. He also provides pseudoconservation laws for two special cases: $p_{ij} = e_{ij}$ with arbitrary number of queues and, two stations with arbitrary p_{ij}'s and e_{ij}'s. Mean waiting times are found for exhaustive and gated service disciplines. System performance can be optimized through playing with these probabilities.

Altiok and Shiue [2] analyze a multi-item (R, r) inventory policy under the assumption of backorders. Production is requested when the inventory in front of the machine drops to r. When there are two such products higher priority products produced first. They approximate the average queue level and backorder level. Their model is simply a polling models with a dynamic polling policy and exhaustive service policy.

Blanc [3] considers a cyclic polling system with Bernoulli service policy. Exponential service times and negligible switchover times are assumed. A method is proposed to find queue length distributions and waiting time distribution without explicitly solving the underlying Markov Chain.

Hofri and Ross [16] investigate the optimal control of two queues where idleness is allowed. Processing times are generally distributed but identical for all queues. They considered two objectives: minimizing the sum of discounted holding cost and switching cost, and long-run average of the sum of holding and switching costs. They construct a Markov decision process where decision epochs are made in service completion, switch completions and arrival points when the server is idle. They show that optimal policy is exhaustive for discounted cost criterion. They further conjectured that this result is natural because service times are identical for all customers. They also showed that server should not switch unless one of the queue length reaches a threshold

value. They also show that such a double threshold policy is optimal for average long run cost criterion. And finally an algorithm is proposed to find these threshold values.

Campbell [7] reviews some of the important works and summarizes some of the important results reported in the literature. He also clarifies the distinction between cyclic server models (polling models) and cyclic customer models (closed queuing networks).

2.1.2 Finite Queue Polling Models

Finite queue polling models have attracted considerably less researchers. It may be due to two reasons: They are very difficult to analyze exactly, so they require more assumptions and they may have less application area. Nevertheless, a number of finite queue models exists in the literature. In all finite queue models, it is assumed that customers who find the queues full are lost.

Takine et al. [37] give a unified approach to polling models with single capacity buffers. They study both exhaustive and gated service policies. Mukherjee et al. [24] improve the work of Takine et al. [37]. They construct a Markov chain embedded at the time instants when stations are polled. Instead of solving all equations, they propose an iterative algorithm.

Tran-Gia and Raith [39] analyze a multi-queue system where arrivals are Poisson, service times and switchover times are generally distributed. But switchover times are queue dependent. Furthermore polling policy is cyclic and service policy is non-exhaustive. They construct an embedded Markov chain. While doing so they find the mean and variance of the cycle time of each queue, and approximate it with a two stage phase type distribution. They also test their algorithm via the computer simulations.

Takagi [35] gives an exact analysis of polling models where arrivals are Poisson and service and switchover times are generally distributed. It is the first work that gives a unifying approach to finite queue polling models. He

considers three service policies: Exhaustive, gated and G-limited. He uses the results of earlier works on single server, finite capacity vacation systems. These works begin with constructing a Markov Chain embedded at service completions and visit beginnings. So, he decomposes the system by assigning vacations when the server is processing the other queues. His approach requires a considerable amount of numerical integrations and inverse Laplace-Stieltjes transforms. The computational tractability of that approach is very low, and hence numerical results are not reported.

Tran-Gia [38] presents an approximate algorithm for polling systems with finite queues, cyclic polling and 1-limited service. All distributions are general but discrete. Therefore, convolutions are made using the fast Fourier transform. Although his model is applicable for nonsymmetric models, the analysis is made for symmetric load conditions. The approximation accuracy is tested with computer simulations.

Albores and Bocharov [1] consider two finite queue with relative priorities, where interarrival and service time distributions have general phase type representations. Switchover times are assumed to be zero. They consider three service disciplines: FIFO, LIFO and RO. Their result is a matrix-algorithmic approach to solve the steady state probabilities.

There are also finite queue optimization models. For example Suk and Cassandras [32] consider the optimal scheduling of two queues competing for one server. There is no switchover time and interarrival and service times are exponentially distributed, furthermore idleness is not allowed. They find that the policy minimizing the total discounted blocking cost and inventory holding cost is of a switching type when the blocking cost is greater than the unit inventory holding cost. Switching type means customer type is served depends on the number of customers waiting for the service.

Rosberg and Kermani [27] deal with scheduling N queues on a single server that maximizes the sum of weighted thruputs of the customers. Arrival process is assumed to be Poisson and service times are exponential. Furthermore switchover times are assumed to be zero. They first find an upper bound to

the value function by decomposing the system and propose two approximate schedules. They also test the performance of their approximate schedules.

Works on finite queues polling models are scarce in the literature. Also there is little implications about the solution techniques. But there is no work that exactly matches with our models which will be described and solved in the next chapter.

Chapter 3

Two Queues M/M/1 Polling Models

3.1 Introduction

As we see from the previous chapter, most of the works in the literature deals with polling systems, with infinite queue capacities. But it is easily seen that there is a lack in finite queues even in Markovian systems, where all the distributions are exponential or of phase type. In this chapter various polling policies are studied under Markovian distributions. Before going into details, it will be beneficial to state the major assumptions:

. All the random variables are assumed to be exponential,

. Switchover times are sequence dependent rather than queue dependent,

. Idleness is not allowed,

. Polling policies are deterministic and cyclical.

. Service disciplines in queues are FIFO.

. Buffer sizes are finite.

. Customers finding full buffer are lost,

. There are two types of customers,

. Preemption is not allowed,

. Server is assumed to be reliable and always ready to operate.

In this study three service policies are considered, namely exhaustive, gated and G-limited.

3.1.1 Notation

Since the same notation is used throughout the thesis, it would be beneficial to give the common notation at the very beginning.

λ_j is the rate of the arrival process of customer j, $j \in J = \{1,2\}$,

Y_j is a random variable (r.v.) representing the processing time of customer $j \in J$, exponential with rate μ_j,

X_{ij} is a r.v. representing switchover time that customer exercises going from customer i to customer j, exponential with rate β_{ij}, $i,j \in J$ and $i \neq j$,

S_j is the capacity of the queue that customer type j joins (excluding the one in the service, if this queue is being served).

$\{N_j(t), t \geq 0\}$ is a stochastic process that represents the number of customers type j, in its queue at time t, where $N_j(t) \in E_j = \{0,1,\ldots,S_j\}$.

$\{Z(t), t \geq 0\}$ is a stochastic process representing the state of server at time t, where $Z(t) \in E_Z = \{1,2,3,4\}$, and

$$Z(t) = \begin{cases} 1 & \text{server processing customer 1} \\ 2 & \text{server processing customer 2} \\ 3 & \text{server switching from customer 1 to customer 2} \\ 4 & \text{server switching from customer 2 to customer 1} \end{cases}$$

Furthermore,

p_{jm} represents the steady state probability of m customers being in queue j,

TH_j is the steady state throughput rate of customer j,

3.1.2 Organization

In this chapter the steady state performance measures of polling systems is found under mainly three service policies. First section is devoted to exhaustive service policy. In this section a Markov Process is defined and its embedded Markov Chain is constructed to calculate the steady state probabilities. Also system is decomposed into two sub-systems that have single server and single customer type, defining imaginary vacations on servers. Furthermore the second approach is simplified in order to get an approximation. Numerical results concerning the approximation accuracy and computational savings gained from decomposition will be presented. The succeeding chapters are devoted to gated and G-limited service policies with the same steps.

3.2 Exhaustive Service Policy

3.2.1 Exact Model

As explained previously, in exhaustive service policy, server alternates between queues whenever the server finds no part in the queue that is currently being served. So the stochastic process $\{N_j(t), Z(t), t \geq 0, j \in J\}$ is a Markov process with the state space $E = E_1 X E_2 X E_Z$, where $|E| = 4(S_1 + 1)(S_2 + 1)$. A particular instance (i_1, i_2, k) means that there is i_1 number of customers in queue 1, i_2 number of customers in queue 2, and server is in state $k \in \{1, 2, 3, 4\}$. Underlying Markov Chain can easily be constructed with the following state

transition rates:

$$
\begin{aligned}
P(i_1,i_2,k)(i_1+1,i_2,k) &= \lambda_1 && \text{for} \quad i_1 \neq S_1, i_2 \in E_2, k \in E_Z,\\
P(i_1,i_2,k)(i_1,i_2+1,k) &= \lambda_2 && \text{for} \quad i_2 \neq S_2, i_1 \in E_1, k \in E_Z,\\
P(i_1,i_2,1)(i_1-1,i_2,1) &= \mu_1 && \text{for} \quad i_1 \neq 0, i_2 \in E_2,\\
P(0,i_2,1)(0,i_2,3) &= \mu_1 && \text{for} \quad i_2 \in E_2,\\
P(i_1,i_2,2)(i_1,i_2-1,2) &= \mu_2 && \text{for} \quad i_2 \neq 0, i_1 \in E_1,\\
P(i_1,0,2)(i_1,0,4) &= \mu_2 && \text{for} \quad i_1 \in E_1,\\
P(i_1,i_2,3)(i_1,i_2-1,2) &= \beta_{12} && \text{for} \quad i_1 \in E_1, i_2 \geq 1,\\
P(i_1,0,3)(i_1,0,4) &= \beta_{12} && \text{for} \quad i_1 \in E_1,\\
P(i_1,i_2,4)(i_1-1,i_2,1) &= \beta_{21} && \text{for} \quad i_2 \in E_2, i_1 \geq 1,\\
P(0,i_2,4)(0,i_2,3) &= \beta_{21} && \text{for} \quad i_2 \in E_2.
\end{aligned}
$$

where $P(i_1,i_2,k)(i_1',i_2',k')$ represents the entry of the infinitesimal generator for which the process goes from state (i_1,i_2,k) to state (i_1',i_2',k').

The construction of the infinitesimal generator will be explained in more detail. The first two equations state that new arrivals join their queues if they are not full with their arrival rates. The next two state that there are departures from the queue 1 with the completion of a service. In the first one server continues processing if there is at least one customer in queue 1, in the second one the server goes to switchover if there is no customer left in the queue 1. The next two simply same for queue 2. The last four equations are about the switchovers. In the first one server begins processing queue 2 after he exercises a switchover with the rate of β_{12}. But if there is no customer in queue 2 it exercises a second switchover to return to queue 1. Last two are also similar.

After constructing the infinitesimal generator, P, it is trivial to find the steady state distribution of queue length and average throughput rates. Since all the states communicate, chain is irreducible and positive recurrent, the steady state distribution of states exists.

Let, $r(i_1,i_2,k)$ be the steady state probability of state (i_1,i_2,k), and $\mathbf{r}$ be the corresponding vector. So, the solution to the following system of equations,

$$\mathbf{r}P = 0 \tag{3.1}$$

$$\sum_{\mathbf{r}\in E} \mathbf{r} = 1 \tag{3.2}$$

provides the required probabilities. Average throughput rates and queue length distributions can also be obtained easily, as follows:

$$TH_j = \mu_j \sum_{n_1,n_2} r(n_1, n_2, j) \tag{3.3}$$

$$p_{1m} = \sum_{n_2,k} r(m, n_2, k) \tag{3.4}$$

$$p_{2m} = \sum_{n_1,k} r(n_1, m, k) \tag{3.5}$$

If one stores all the elements of the rate matrix P, memory requirements will be $O(|E|^2)$, and computation time will be about $O(|E|^3)$, to solve steady state probabilities by Gaussian Elimination. But rate matrix is quite sparse, approximately three nonzeros per row. Therefore the sparsity is exploited using a sparse solver with enormous savings in both computation and memory requirements. The use of sparse solver will be explained in detail in numerical computation section.

3.2.2 Decomposition

The aim of the decomposition is to reduce the problem into a manageable size. The approach is a very simple and widely used. The system is approximated with two one-server queues, defining new pseudo-servers to represent the behavior of the original server. To accomplish this task, vacations are scheduled for the pseudo-servers in order to model the switching of the original server. When a buffer becomes empty, a vacation is scheduled for its pseudo-server. The primary task is to determine the distributions of these vacations, which will be explained in the following subsection. Figure 3.1 can help to better understand the decomposition methodology.

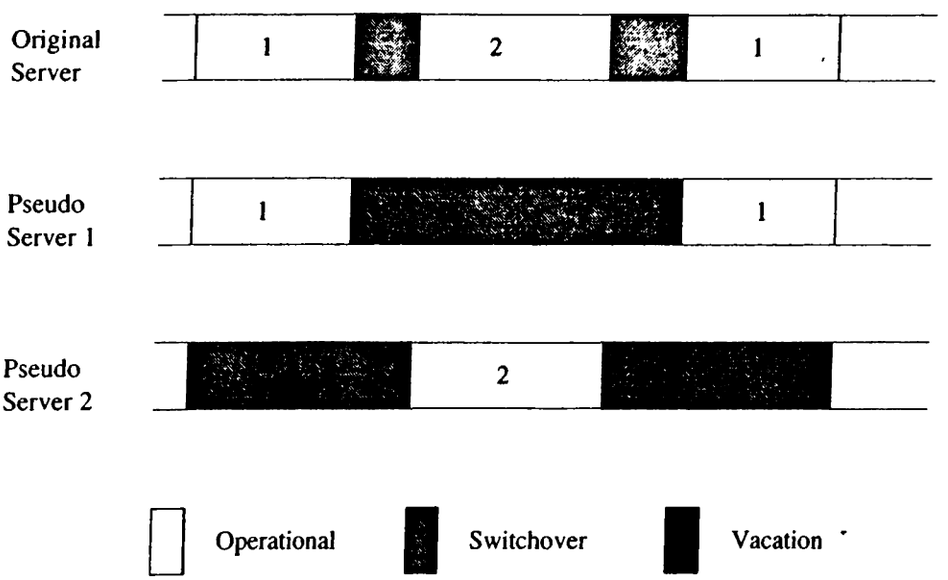

Figure 3.1: An Example of Gantt Charts

Vacation Periods

When server finishes serving a queue in a cycle, it switches to the other one. Now, vacation for the pseudo-server begins. A switchover time elapses, while server is set up for the other type of customer. Server turns back after it finishes all the jobs in the second queue. Server begins to serve the first queue after a second switchover time is also elapsed. This is end of the vacation of the pseudo-server as well as the beginning of a new cycle. Same process applies for the second queue. Let

$\eta_j^{(l)}$ be a r.v. denoting the vacation period of pseudo-server j, in cycle l.

So, it is clear that:

$$\eta_1^{(l)} = X_{12} + \gamma_2^{(l)} + X_{21} \tag{3.6}$$

$$\eta_2^{(l)} = X_{21} + \gamma_1^{(l)} + X_{12} \tag{3.7}$$

where $\gamma_j^{(l)}$ is a r.v. denoting the time required to finish all customers in queue j, in cycle l. It will be called as *clearance time*.

The distributions of ξ_{12} and ξ_{21} are known. Primary task is to find exact

distributions for $\gamma_j^{(l)}$. If $\gamma_j^{(l)} \to \gamma_j$ as $l \to \infty$ then it is obvious that $\eta_j^{(l)} \to \eta_j$ as $l \to \infty$. The existence for the invariant distribution of clearance times will be questioned later in this section. For the time being it is assumed that the distribution exists.

Let $\gamma_j^{(k)}$ be the *clearance time* when the server visits queue j and faces with k customers in the queue, then it is obvious that:

$$\gamma_j = \begin{cases} \gamma_j^{(0)} & \text{with probability } q_{j0} \\ \gamma_j^{(1)} & \text{with probability } q_{j1} \\ \vdots & \\ \gamma_j^{(k)} & \text{with probability } q_{jk} \\ \vdots & \\ \gamma_j^{(S_j)} & \text{with probability } \tilde{q}_{jS_j} \end{cases} \tag{3.8}$$

where q_{jk} is the probability of being k customers in queue j, when the server is ready for the processing customer type j, and $\tilde{q}_{jS_j} = \sum_{k=S_j}^{\infty} q_{jk}$. Notice that there cannot be more than S_j customers, independent of the number of arrivals during the vacation time.

Suppose that there are k customers at a time. If a new arrival occurs before the first service completed (with probability $\lambda_j/(\mu_j + \lambda_j)$) then clearance time will be as if there are $(k+1)$ customers. This follows from the memoryless property of the exponential distribution. If the service is completed before a new arrival (with probability $\mu_j/(\mu_j + \lambda_j)$) then clearance time will be sum of the service time of the customer (Y_j) and clearance time as if there are $(k-1)$ customers. With this idea following system of linear equalities is derived as follows:

$$\begin{aligned} \gamma_j^{(0)} &= 0 \\ \gamma_j^{(1)} &= \mu_j/(\mu_j + \lambda_j)(\gamma_j^{(0)} + Y_j) + \lambda_j/(\mu_j + \lambda_j)\gamma_j^{(2)} \\ \gamma_j^{(2)} &= \mu_j/(\mu_j + \lambda_j)(\gamma_j^{(1)} + Y_j) + \lambda_j/(\mu_j + \lambda_j)\gamma_j^{(3)} \end{aligned}$$

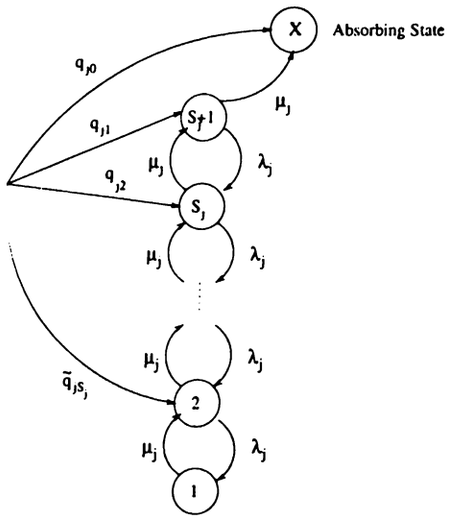

Figure 3.2: Phase-type Representation of $\gamma_j \quad j \neq k$

$$\gamma_j^{(k)} = \mu_j/(\mu_j+\lambda_j)(\gamma_j^{(k-1)}+Y_j) + \lambda_j/(\mu_j+\lambda_j)\gamma_j^{(k+1)}$$

$$\gamma_j^{(S_j)} = \mu_j/(\mu_j+\lambda_j)(\gamma_j^{(S_j-1)}+Y_j) + \lambda_j/(\mu_j+\lambda_j)\gamma_j^{(S_j)}$$

If one takes the expectation of both sides in above system of linear equalities, a system of linear equations can be obtained, where there are (S_j+1) variables and (S_j+1) equations. $E(\gamma_j^{(k)})$ for $k=0,...,S_j$ can be obtained easily. In general, it is possible to get the Laplace transform of the $\gamma_j^{(k)}$s. In the above argument all the properties of exponential distribution are not exploited, instead there is a more rigorous method to find the distribution functions of not only clearance times but also vacation periods.

A phase-type distribution can be defined with a Markov process, in which all states are transient except the absorbing state and an initial probability vector. Figure 3.2 represents the states of the distribution of γ_j. States 1 through S_j+1 represents the number of customers in the system in reverse order. State **X** is the absorbing state. Therefore clearance time of pseudo-machine j is the time elapsed the process reaches to state **X**. Initial probability vector of this distribution is $(0, \tilde{q}_{jS_j}, q_{jS_j-1}, \ldots, q_{j1}, q_{j0})$. As a result if q_{jk}'s are

estimated, complete description of γ_j can be obtained. Notice that vacation period is nothing but a convolution of two switchovers and the clearance time of the other queue. Neuts [25] states that convolution of phase-type distributions are also phase-type.

If vacation period of pseudo-server 2 is given, i.e., T, then it is obvious that

$$\begin{aligned} q_{2k} &= e^{-\lambda_2 T}(\lambda_2 T)^k/k!, k = 0, 1, ..N_2 - 1, \\ \tilde{q}_{2S_2} &= \sum_{k=S_2}^{\infty} e^{-\lambda_2 T}(\lambda_2 T)^k/k!. \end{aligned}$$

which corresponds to Poisson distribution. But vacation period of each pseudo-servers is a random variable that has phase-type representation. So, generalization easily follows,

$$\begin{aligned} q_{jk} &= \int_0^{\infty} (e^{-\lambda_j u}(\lambda_j u)^k/k!) f_{\eta_j}(u)du, k = 0, 1, ..S_j - 1, \\ \tilde{q}_{jS_j} &= \sum_{k=S_j}^{\infty} \int_0^{\infty} (e^{-\lambda_j u}(\lambda_j u)^k/k!) f_{\eta_j}(u)du, \end{aligned} \tag{3.9}$$

As it is noted earlier η_j has a phase-type representation as it is shown in Figure 3.3. Let Q_j be the infinitesimal generator of corresponding Markov Chain, with the state space, $\{1, 2, \ldots, S_j + 3, \mathbf{X}\}$. As Neuts [25] states Q_j has the following form:

$$Q_j = \begin{bmatrix} A_j & A_j^0 \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.10}$$

where the $(S_j + 3)$ X $(S_j + 3)$ matrix A_j has negative diagonal and positive off-diagonal entries. Also $A_j\mathbf{e} + A_j^0 = \mathbf{0}$, and the initial probability vector of Q_j is given by $(\mathbf{q_j}, q_x)$, with $\mathbf{q_j e} + q_x = 1$. But all states except state $\mathbf{X}$ should be transient, that is absorption into the state $\mathbf{X}$ should be certain. Sufficient condition is simple : $\lambda_j < \mu_j$. Now it is time to determine A_j and the initial probability vector $\mathbf{q_j}$.

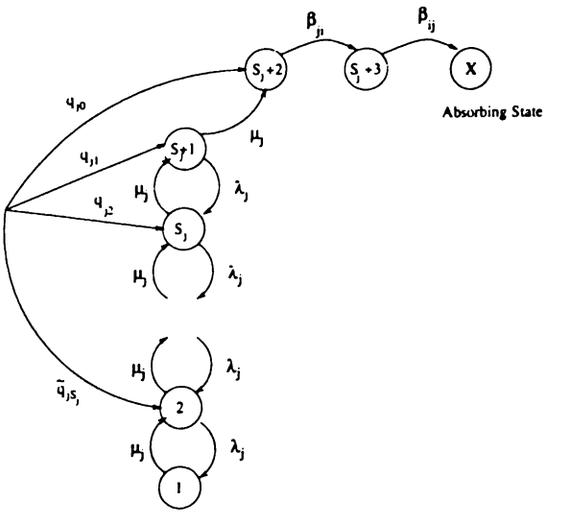

Figure 3.3: Phase-type Representation of $\eta_j \quad j \neq k$

The matrix A_j, that is nothing but the infinitesimal generator of the Markov Process, is:

$$A_j = \begin{bmatrix} -\mu_i & \mu_i & 0 & & 0 & 0 & 0 & 0 \\ \lambda_i & -(\lambda_i+\mu_i) & \mu_i & & 0 & 0 & 0 & 0 \\ 0 & \lambda_i & -(\lambda_i+\mu_i) & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -(\lambda_i+\mu_i) & \mu_i & 0 & 0 \\ 0 & 0 & 0 & & \lambda_i & -(\mu_i+\lambda_i) & \mu_i & 0 \\ 0 & 0 & 0 & & 0 & 0 & -\beta_{ij} & \beta_{ij} \\ 0 & 0 & 0 & & 0 & 0 & 0 & -\beta_{ji} \end{bmatrix},$$

where $i \neq j$. Notice that the switchovers are added as (S_j+2)nd and (S_j+3)rd states. The initial probability vector is:

$$\mathbf{q_j} = (0, \tilde{q}_{jS_j}, q_{jS_j-1}, \dots, q_{j1}, q_{j0}, 0)$$

As Neuts [25] states, $f_{\eta_j}(u) = \mathbf{q_{i\neq j}} \exp(A_j u) A_j^0$. This is the crucial point to find $\mathbf{q_j}$s, because system of equations (3.9) reduce to a system of linear equations with S_1+S_2+2 equations and S_1+S_2+2 unknowns. From the above

arguments it is clear that there exists stationary distribution for clearance times if $q_{jk}^{(l)} \rightarrow q_{jk}$ as $l \rightarrow \infty$. Let us rewrite equations 3.9 by incorporating the cycle index,

$$\begin{aligned} q_{jk}^{(l+1)} &= \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{q_i^{(l)}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad \text{for } i,j \in J,\ i \neq j,\ \ k = 0,1,..S_j - 1, \\ \tilde{q}_{jS_j}^{l+1} &= \sum_{k=S_j}^{\infty} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{q_i^{(l)}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \text{ for } i,j \in J,\ i \neq j, \end{aligned}$$

which reduce to

$$\begin{aligned} q_{jk}^{(l+1)} &= \sum_{m=0}^{S_i-1} q_{im}^{(l)} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{e_{im}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad + \tilde{q}_{iS_i}^{(l)} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{e_{iS_i}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad \text{for } i,j \in J,\ i \neq j,\ \ k = 0,1,..S_j - 1, \\ \tilde{q}_{jS_j}^{l+1} &= \sum_{k=S_j}^{\infty} \sum_{m=0}^{S_i} q_{im}^{(l)} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{e_{im}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad + \sum_{k=S_j}^{\infty} \tilde{q}_{iS_i}^{(l)} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{e_{iS_i}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad \text{for } i,j \in J,\ i \neq j, \end{aligned} \tag{3.11}$$

where $\mathbf{e_{im}}$ is a row vector of appropriate size that has all zeros except the entry corresponding to q_{im}^l, which is one. Let,

$$\begin{aligned} a_{jmk} &= \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!)\mathbf{e_{im}} \exp(\mathbf{A_j}u)\mathbf{A_j^0}du, \\ &\quad \text{for } i,j \in J,\ i \neq j,\ \ k = 0,1,..,S_j - 1,\ m = 0,1,...S_i, \text{ and} \\ \tilde{a}_{jmS_j} &= \sum_{k=S_j}^{\infty} a_{jmk}, \text{ for } j \in J, \text{ and } m = 0,1,...S_i. \end{aligned}$$

Then Equations 3.11 become :

$$q_{jk}^{(l+1)} = \sum_{m=0}^{S_i-1} q_{im}^{(l)} a_{jmk} + \tilde{q}_{iS_i} a_{jS_ik}, \text{ for } i,j \in J,\ i \neq j,\ \ k = 0,1,...,S_j - 1, \text{ and,}$$

$$\dot{q}_{jS_j}^{(l+1)} = \sum_{m=0}^{S_i-1} q_{im}^{(l)} \tilde{a}_{jmS_j} + \tilde{q}_{iS_i} \tilde{a}_{jS_iS_j}, \text{ for } i,j \in J,\ i \neq j, \text{ and,} \tag{3.12}$$

Therefore aforementioned linear system is obtained. Following theorem states the existence of limiting q_{jk}'s.

Theorem 3.1 $\mathbf{q_j^{(l)} \to q_j}$ *as* $l \to \infty$ *for* $j \in J$.

Proof: Let,

$$T_j = \begin{bmatrix} a_{j00} & a_{j10} & \cdots & a_{jm0} & & a_{jS_i0} \\ a_{j01} & a_{j11} & \ddots & a_{jm1} & & a_{jS_i1} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{j0k} & a_{j1k} & & a_{jmk} & \cdots & a_{jS_ik} \\ \vdots & \vdots & & \vdots & & \vdots \\ \tilde{a}_{j0S_j} & \tilde{a}_{j1S_j} & \cdots & \tilde{a}_{jmS_j} & \cdots & \tilde{a}_{jS_iS_j} \end{bmatrix},$$

Above system of equations 3.12 can be rewritten in matrix form as:

$$\begin{bmatrix} \mathbf{q_1^{(l+1)}} \\ \mathbf{q_2^{(l+1)}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{T_1} \\ \mathbf{T_2} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{q_1^{(l)}} \\ \mathbf{q_2^{(l)}} \end{bmatrix} \tag{3.13}$$

Notice that,

$$\sum_{k=0}^{S_j-1} a_{jmk} + \tilde{a}_{jmS_j} = 1, \text{ for } i,j \in J,\ i \neq j, \text{ and } m = 0,1,\ldots,S_i.$$

That is the sum of the columns are one. Hence, the matrix in Equation 3.13 is a Markov matrix. Since $a_{jmk} > 0$ for all $j \in J$, and $m = 0,1,\ldots,S_i$ all states are communicating and chain is positive recurrent and all states has period two. Due to Çınlar [9] there exist stationary probabilities satisfying $\sum_{k=1}^{S_j-1} q_{jk} + \dot{q}_{jS_j} = 1$ for $j \in J$. □

As a result, the existence of stationary vacation period distributions is proved the complete description of the system is obtained. But, finding the exact parameters of the Equations 3.12 requires excessive amount of calculations. It is also possible to have an ill-conditioned system, because the magnitudes of the entries can be in very different scales. Instead, an iterative algorithm is proposed to find q_{jk}'s. Before going into details of the algorithm, a theorem will be presented. This theorem gives a simple way of calculating the q_{jk}'s. The proof can be found in Neuts [25].

Theorem 3.2 *Let* $a_k = \int_0^\infty e^{-\lambda u}(\lambda u)^k/k! dF(u)$ *for* $k \geq 0$, *and* $F(u)$ *is a phase type distribution with representation* $(\alpha, \mathbf{T})$, *where* α *is the initial probability vector and* T *is the rate matrix, then* $\{a_k\}$ *is a discrete phase type density with representation* $(\beta, \mathbf{S})$, *given by*

$$\beta = \lambda\alpha(\lambda\mathbf{I} - \mathbf{T})^{-1}, \qquad \mathbf{S} = \lambda(\lambda\mathbf{I} - \mathbf{T})^{-1}, \tag{3.14}$$

Neuts also provides an efficient way of computing the density $\{a_k\}$ by evaluating vectors

$$\nu(0) = \alpha/\lambda, \qquad \nu(k+1) = \nu(k)\mathbf{S}, \quad for \quad k \geq 0, \tag{3.15}$$

and

$$a_0 = \nu(1)T^0, \text{ and } a_k = \nu(k+1)T^0, \quad for \quad k \geq 0,$$

also,

$$\sum_{i=k+1}^{\infty} a_i = \lambda\nu(k+1)\mathbf{e}, \quad for \quad k \geq 0.$$

where $\mathbf{e}$ is a column vector of ones.

An iterative algorithm is given below to find the q_{jk}'s using the above results. The algorithm is a fixed point algorithm, which stops after differences between two successive iterations are less than a fixed and predetermined ϵ value. If this criteria is not satisfied it it terminated after a fixed number of

iterations are performed. These probabilities make possible to find the vacation periods of pseudo-servers as well as help to model the decomposed systems as Markov processes.

Algorithm 3.1

Step:=0;
approximate:=false;
$q_{jk}^0 := 1/(S_j + 1)$: *for* $j = 0, 1, \ldots, S_j$, $j \in J$
while not *approximate* **and** *(Step* $\leq$ *MaxStep)* **do**
  $v_2 = \mathbf{q_2^{Step}}/\lambda_1$
  for $k = 0, k < S_1$ **do**
    $v_2 = v_2\lambda_1(\lambda_1 I - T_2)^{-1}$
    $q_{1k}^{Step+1} = v_2 T_2^0$
  $\tilde{q}_{1S_1}^{Step+1} = \lambda_1 v_2 e$
  $v_1 = \mathbf{q_1^{Step+1}}/\lambda_2$
  for $k = 0, k < S_2$ **do**
    $v_1 = v_1\lambda_2(\lambda_2 I - T_1)^{-1}$
    $q_{2k}^{Step+1} = v_1 T_1^0$
  $\tilde{q}_{2S_2}^{Step+1} = \lambda_2 v_1 e$
  if $\frac{|q_{2k}^{Step+1} - q_{2k}^{Step}|}{|q_{2k}^{Step+1}|} \leq \epsilon$ *for each* k **then**
    approximate:=true;
  Step:=Step+1;
if *approximate=true* **then**
  set q_{jk}^{Step} *as steady state probabilities for each* k *and* j *and* **return** *Step*
else
  return *a message of failure*
end.

In computing $v_2 = v_2\lambda_1(\lambda_1 I - T_2)^{-1}$ and $v_1 = v_1\lambda_2(\lambda_2 I - T_1)^{-1}$ explicit inverses of the matrices are not used, rather v_j's are solved by Gaussian elimination. Since the matrices $(\lambda_1 I - T_2)$ and $(\lambda_2 I - T_1)$ are tridiagonal, the

computational complexity is $O(S_j)$ operations. For one step there are $O(S_1 S_2)$ operations. Furthermore, no row interchanges are required because matrices are diagonally dominant.

Algorithm 3.1 simply solves the system of equations 3.13 recursively in the following manner:

$$\mathbf{q_1^{(l+1)}} = \mathbf{T_1 q_2^{(l)}}, \text{ and,} \tag{3.16}$$

$$\mathbf{q_2^{(l+1)}} = \mathbf{T_2 q_1^{(l+1)}}, \text{ and,} \tag{3.17}$$

Following theorem states the convergence of the Equations 3.16 and 3.17, thus convergence of the Algorithm 3.1.

Theorem 3.3 *Equations 3.16 and 3.17 converges.*

Proof: The same idea in the proof of Theorem 3.1 will be used. Let us rewrite the equations 3.16 and 3.17 as,

$$\mathbf{q_j^{(l+1)}} = \mathbf{T_j T_i q_j^{(l)}}, \text{ for } i,j \in J \; i \neq j. \tag{3.18}$$

where

$$T_j T_i = \begin{bmatrix} \sum_{m=0}^{S_i-1} a_{jm0} a_{i0m} + a_{jS_i0}\tilde{a}_{i0S_i} & \sum_{m=0}^{S_i-1} a_{jm0} a_{i1m} + a_{jS_i0}\tilde{a}_{i1S_i} \\ \sum_{m=0}^{S_i-1} a_{jm1} a_{i0m} + a_{jS_i1}\tilde{a}_{i0S_i} & \sum_{m=0}^{S_i-1} a_{jm1} a_{i1m} + a_{jS_i1}\tilde{a}_{i1S_i} \\ \vdots & \vdots \\ \sum_{m=0}^{S_i-1} \tilde{a}_{jmS_j} a_{i0m} + \tilde{a}_{jS_iS_j}\tilde{a}_{i0S_i} & \sum_{m=0}^{S_i-1} \tilde{a}_{jmS_j} a_{j1m} + \tilde{a}_{jS_iS_j}\tilde{a}_{i1S_i} \end{bmatrix}$$

$$\begin{bmatrix} \sum_{m=0}^{S_i-1} a_{jm0} a_{jS_jm} + a_{jS_i0}\tilde{a}_{iS_jS_i} \\ \sum_{m=0}^{S_i-1} a_{jm1} a_{jS_jm} + a_{jS_i1}\tilde{a}_{iS_jS_i} \\ \vdots \\ \cdots \quad \sum_{m=0}^{S_i-1} \tilde{a}_{jmS_j} a_{iS_jm} + \tilde{a}_{jS_iS_j}\tilde{a}_{iS_jS_i} \end{bmatrix}.$$

Notice that

$$
\begin{aligned}
& \sum_{k=0}^{S_j-1}\Big(\sum_{m=0}^{S_i-1} a_{jmk}a_{inm} + a_{jS_ik}\tilde{a}_{inS_i}\Big) + \sum_{m=0}^{S_i-1} \tilde{a}_{jmS_j}a_{inm} + \tilde{a}_{jS_iS_j}\tilde{a}_{inS_i} \\
= & \sum_{m=0}^{S_i-1}\Big(\sum_{k=0}^{S_j-1} a_{jmk}a_{inm} + \tilde{a}_{jmS_j}a_{inm}\Big) + \sum_{m=0}^{S_i-1} a_{jS_ik}\tilde{a}_{inS_i} + \tilde{a}_{jS_iS_j}\tilde{a}_{inS_i}, \\
= & \sum_{m=0}^{S_i-1} a_{inm}\Big(\sum_{k=0}^{S_j-1} a_{jmk} + \tilde{a}_{jmS_j}\Big) + \tilde{a}_{inS_i}\Big(\sum_{k=0}^{S_j-1} a_{jS_ik} + \tilde{a}_{jS_iS_j}\Big), \\
= & \sum_{m=0}^{S_i-1} a_{inm}(1) + \tilde{a}_{inS_i}(1) \\
= & \ 1, \text{ for } n = 0, 1, ..., S_j.
\end{aligned}
$$

That is the sum of the columns adds up to one. So, the matrices (T_1T_2) and (T_2T_1) are Markov matrices. Since $a_{jmk} > 0$ for all $j \in J$, and $k = 0, 1, \ldots, S_j, m = 0, 1, \ldots, S_i$ and all states are communicating the chains are ergodic. □

Hence, the stationary probabilities exist, resulting the algorithm finds the steady state probabilities of a Markov chain as if multiplying matrices by themselves until difference between probabilities at successive steps becomes insignificant. Thus, the Algorithm 3.1 converges. The convergence rate of this algorithm can be found in Çınlar [9].

Decomposed Models

Let the stochastic process $\{Z_j(t), t \geq 0\}, j \in J$ representing the state of the pseudo-server j where $Z_j(t) \in E_{Z_j} = \{1, 2, \ldots, S_i + 4\}$, $i \in J$ and $i \neq j$, such that

$$Z_j(t) = \begin{cases} 1 & \text{pseudo-server } j \text{ processing customer } j \text{ at time } t, \\ 2 & \text{pseudo-server } j \text{ is at vacation at time } t, \text{ (original server is in switchover),} \\ 2+k & \text{pseudo-server } j \text{ is at vacation at time } t, k = 1,2,\ldots,S_i+1, \\ & i \neq j, \text{ (original server is clearing the other queue),} \\ S_i+1 & \text{pseudo-server } j \text{ is at vacation at time } t \text{ (original server is in switchover to return back),} \end{cases}$$

Therefore, we can model the one server models as stochastic processes $\{N_j(t), Z_j(t), t \geq 0\}, j \in J$. A particular instance (i,k) means that there is i number of customers in queue j, and pseudo-server j is in state k. Since vacation periods are of phase type, these processes are Markov processes with state space $E'_j = E_j X E_{Z_j}$ where $|E'_1| = (S_1+1)(S_2+4)$ and $|E'_2| = (S_2+1)(S_2+4)$.

Markov Chains of the above Markov processes can be easily constructed as follows:

$$\begin{aligned}
P_1(i,k)(i+1,k) &= \lambda_1 && \text{for} \quad i \neq S_1, k \in E_{Z_1}, \\
P_1(i,1)(i-1,1) &= \mu_1 && \text{for} \quad i \geq 1, \\
P_1(0,1)(0,2) &= \mu_1 && \\
P_1(i,2)(i,4) &= \tilde{q}_{2,S_2}\beta_{12} && \text{for} \quad i \in E_1, \\
P_1(i,2)(i,k) &= q_{2,(S_2+4-k)}\beta_{12} && \text{for} \quad i \in E_1, 5 \leq k \leq S_2+4, \\
P_1(i,k)(i,k+1) &= \mu_2 && \text{for} \quad i \in E_1, 3 \leq k \leq S_2+3, \\
P_1(i,k)(i,k-1) &= \lambda_2 && \text{for} \quad i \in E_1, 4 \leq k \leq S_2+3, \\
P_1(i,S_2+4)(i-1,1) &= \beta_{21} && \text{for} \quad i \geq 1, \\
P_1(0,S_2+4)(0,2) &= \beta_{21}, &&
\end{aligned}$$

where $P(i_1,k)(i'_1,k')$ represents the entry of the infinitesimal generator for which the process goes from state (i_1,k) to state (i'_1,k').

The construction of the infinitesimal generator will be explained in more detail. P1 is the generator of the Markov process of the system where pseudo-server 1 is operating. The first equation states that new arrivals join queue 1 with rate λ_1. if the queue is not full.The next one states that there are departures from the queue 1 with the completion of a service. Third equation

states that, server goes for a switchover after a service completion if there is no customer left to processed in the queue. This the beginning of the vacation. The next for equations state the transition of the states for which the the other queue is processed. If the other queue is cleared, server exercises a switchover to return from the vacation and begins to process queue 1. This is the end of the vacation period. The one before the last states that fact. The last one is the same but, it states that if there is no customer waiting server takes a second vacation.

In the light of the above discussion it is an easy task to generate the infinitesimal generator for the second system as follows:

$$\begin{aligned}
P_2(i,k)(i+1,k) &= \lambda_2 && \text{for} \quad i \neq S_2, k \in E_{Z_2}, \\
P_2(i,1)(i-1,1) &= \mu_2 && \text{for} \quad i \geq 1, \\
P_2(0,1)(0,2) &= \mu_2 && \\
P_2(i,2)(i,4) &= \tilde{q}_{1,S_1}\beta_{21} && \text{for} \quad i \in E_2, \\
P_2(i,2)(i,k) &= q_{1,(S_1+4-k)}\beta_{21} && \text{for} \quad i \in E_2, 4 \leq k \leq S_1+4, \\
P_2(i,k)(i,k+1) &= \mu_1 && \text{for} \quad i \in E_2, 3 \leq k \leq S_1+3, \\
P_2(i,k)(i,k-1) &= \lambda_1 && \text{for} \quad i \in E_2, 4 \leq k \leq S_1+3, \\
P_2(i,S_1+4)(i-1,1) &= \beta_{12} && \text{for} \quad i \geq 1, \\
P_2(0,S_1+4)(0,2) &= \beta_{12}, &&
\end{aligned}$$

After constructing the infinitesimal generators,$P_j, j \in J$, it is trivial to find the steady state distribution of queue length and average throughput rates. Since all the states communicate, chains are irreducible and recurrent.

Let $r_j(i,k)$ be the steady state probability of state (i,k) and $\mathbf{r_j}$ be the corresponding vectors. So system of equations ,

$$\begin{aligned}
\mathbf{r_j} P_j &= 0, \\
\sum_{\mathbf{r_j} \in E_j'} \mathbf{r_j} &= 1, \qquad \text{for} \quad j \in J.
\end{aligned} \tag{3.19}$$

gives the steady state probabilities. Average throughput rates and queue length distributions are obtained easily, as follows:

$$TH_j = \mu_j \sum_i r_j(i,1) \tag{3.20}$$

$$p_{jm} = \sum_{k \in E_{z_j}} r_j(m,k) \tag{3.21}$$

If one stores all the elements of the rate matrices P_j, memory requirements will be $O(|E'_j|^2)$, and computation time will be about $O(|E'_j|^3)$, to solve steady state probabilities by Gaussian Elimination, for each decomposed system as in the exact method. But the rate matrix is quite sparse, approximately three to five nonzeros per row. The sparsity is exploited using a sparse solver with enormous savings in both computation and memory requirements. The use of sparse solver will be explained in detail in Chapter 4.

3.2.3 Approximation

As Takagi [35] has noted the above decomposition is exact as long as the exact distribution of the vacation periods is known. Therefore as long as the probabilities q_{jk}'s are estimated accurately, the exact distribution of vacation periods can be obtained. But decomposed models also require solving the steady state distributions through a set of linear equations. In the remaining of this section an approximation method is devised to find basic performance measures, that do not need to solve the Markov chain matrix.

A *cycle* is defined as the time between, the server's successive visit beginnings to a queue. So, the cycle time has four components: Two clearance times and two switchover times. In this respect cycle time can also be defined as the time between successive visit endings. Throughout the analysis, we use the latter definition of the cycle time. Since the model is cyclical, then the cycle time seen by the both queues are equal. Let C represents the cycle time, then,

$$C = \gamma_1 + \xi_{12} + \gamma_2 + \xi_{21}. \tag{3.22}$$

Since cycle time is a convolution of four phase type distributions, C has also a phase type distribution [25]. If C's first moment exists, the stochastic processes $\{Z(t), t \geq 0\}$ and $\{N_j(t), t \geq 0\}, j \in J$ are *regenerative processes*, with state spaces $\{1, 2, 3, 4\}$, and E_j respectively, because at every visit endings of a particular queue, process restarts itself probabilistically. Consequently, these visit endings constitute the event times of a renewal process having $F_C(u)$ as the interarrival distribution.

Following theorem will be very helpful to find the performance measures. Its proof is given in [26].

Theorem 3.4 *Let P_j be the long run proportion of time, that process $\{Y(t), t \geq 0\}$ is in state j, and if $F_C(u)$ has a density over some interval, and $E(C) < \infty$, then*

$$P_j = lim_{t\to\infty} P\{Y(t) = j\} = \frac{E(amount\ of\ time\ in\ state\ j \quad during\ a\ cycle)}{E(time\ of\ a\ cycle)} \tag{3.23}$$

The system fulfills the assumptions of Theorem 3.4. $F_C(u)$ is continuous and has a density on $(0, \infty)$ and $E(C) < \infty$. Therefore thruput rates are:

$$TH_j = \frac{E(\gamma_j)}{E(C)}, \text{ for } j \in J. \tag{3.24}$$

But with the existing information, it is hard to find the queue length distributions, and mean queue lengths are rather found. The average queue length in a cycle, is the long run average queue length.

A particular realization of the process $\{N_j(t), t \geq 0\}$ in a cycle is given in Figure 3.4. First part in the figure represents the inventory accumulation during the vacation period of pseudo-server j, second part shows the process until queue is cleared. Let, A_j be the area under the process $\{N_j(t), t \geq 0\}$, in a cycle. Then,

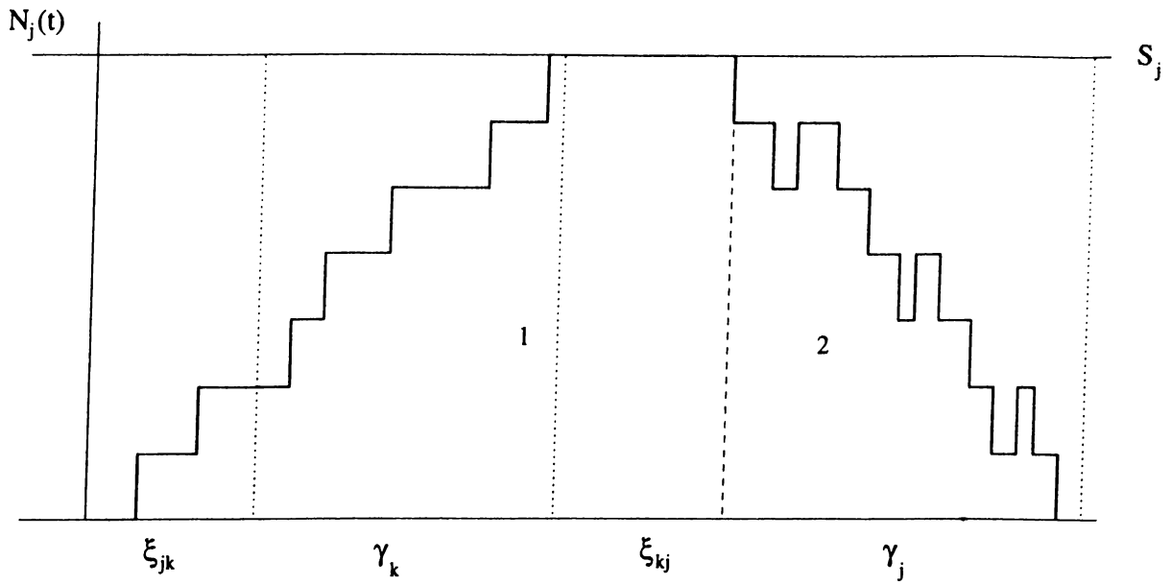

Figure 3.4: Exhaustive Case:A Particular Realization of N_j, $j \neq k$.

$$E[N_j(t)] = E[A_j]/E(C). \tag{3.25}$$

In Figure 3.4 first region represents the total inventory during a vacation period and second region is the total inventory as the queue is cleared. Let A_j^1 and A_j^2 be the total inventories of regions 1 and 2 respectively. First part is analyzed under the assumption of infinite queues. This assumption is relaxed later. Let $\{t_k, k \geq 0\}$ be the arrival points during the vacation period, shown in Figure 3.5. Then,

$$A_j^1 = \sum_{n=1}^{M} (\eta_j - t_n), \text{ for } j \in J \tag{3.26}$$

where M is a r.v. representing the number of arrivals during the vacation period. This expression is nothing but the sum of horizontal bands as shown in Figure 3.5.

The expectation of A_j^1, conditioned on the number of arrivals during the vacation time, and the vacation time itself is:

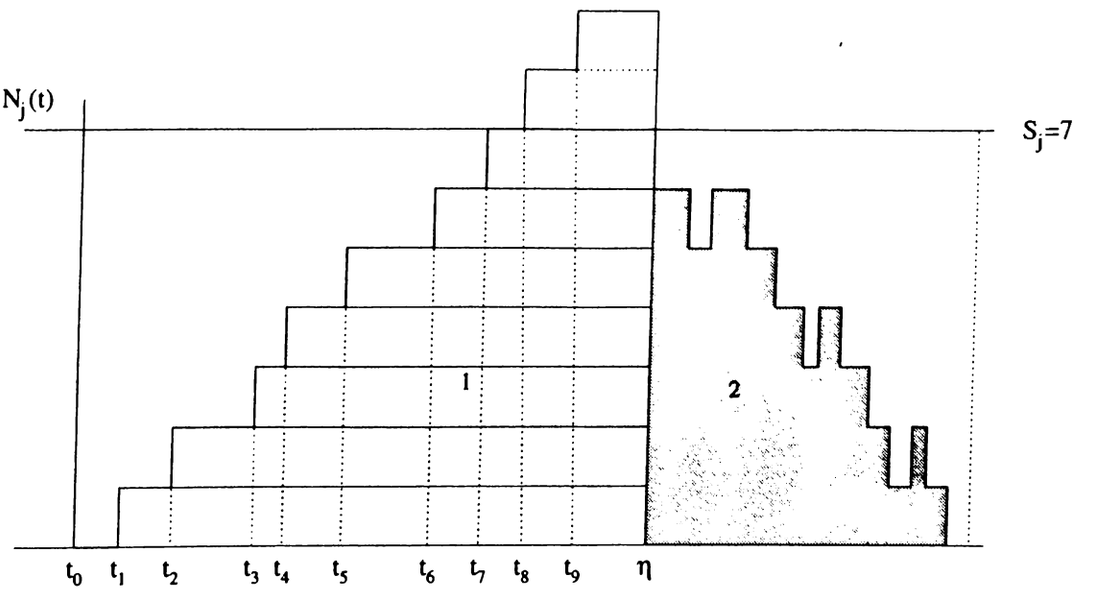

Figure 3.5: Exhaustive Case:Total Inventory Process During the Cycle Time

$$E(A_j^1|N(\eta_j) = m|\eta_j = t) = \sum_{n=1}^{m}(t - t_n) = \frac{tm}{2}, \text{ for } j \in J \tag{3.27}$$

because $\{t_n, n = 1, \ldots, m\}$ is the ordered statistics of uniform variables in $(0, t)$, [19]. The unconditional expectation, is:

$$\begin{aligned} E(A_j^1) &= \int_0^\infty \sum_{m=1}^{\infty} \frac{tm}{2} P(N(\eta_j) = m|\eta_j = t) f_{\eta_j}(t)dt, \text{ for } j \in J & (3.28)\\ &= \int_0^\infty \frac{t}{2} \sum_{m=0}^{\infty} m e^{-\lambda_j t} \frac{(\lambda_j t)^m}{m!} f_{\eta_j}(t)dt, \text{ for } j \in J & (3.29)\\ &= \int_0^\infty \frac{\lambda_j t^2}{2} f_{\eta_j}(t)dt, \text{ for } j \in J & (3.30)\\ &= \frac{\lambda_j E(\eta_j^2)}{2}, \text{ for } j \in J & (3.31) \end{aligned}$$

where $E(\eta_j^2) = 2(\mathbf{q_j A_j^{-2} e})$ where $\mathbf{e}$ is a column of ones of appropriate size.

In the above expression total inventory accumulation is conditioned on the number of arrivals during the vacation time and the vacation time distribution. But the analysis becomes quite complex when the queue capacities are finite.

But, if there are more arrivals than the buffer capacity can hold, the inventory accumulation after the capacity exceeded is subtracted from the infinite queue result. Hence,

$$E(A_j^1|t_{S_j}=u) = \frac{\lambda_j E(\eta_j^2)}{2} - \int_0^\infty \sum_{m=S_j+1}^{\infty} \frac{(m-S_j)(t-u)}{2} P(N(\eta_j)=m|\eta_j=t) f_{\eta_j}(t)dt, \text{ for } j \in J \quad (3.32)$$

where u is a r.v. representing the time of the S_jth arrival.

In the above expression, the analysis becomes very trivial as long as the conditional expectation of t_{S_j} is used instead of u. Because a second integration with the distribution of u is eliminated. Larsson [19] shows that $E(t_{S_j}|N(\eta_j)=m|\eta_j=t)=\frac{tS_j}{m+1}$. Therefore, the above equation reduces to,

$$E(A_j^1) = \frac{\lambda_j E(\eta_j^2)}{2} - \int_0^\infty \sum_{m=S_j+1}^{\infty} \frac{(m-S_j)(t-\frac{tS_j}{m+1})}{2} P(N(\eta_j)=m|\eta_j=t) f_{\eta_j}(t)dt, \text{ for } j \in J \quad (3.33)$$

$$= \frac{\lambda_j E(\eta_j^2)}{2} - \sum_{m=S_j+1}^{\infty} (\frac{m-S_j}{2\lambda_j})(m+1-S_j) \int_0^\infty e^{-\lambda_j t} \frac{(\lambda_j t)^{m+1}}{(m+1)!} f_{\eta_j}(t)dt, \text{ for } j \in J \quad (3.34)$$

$$= \frac{\lambda_j E(\eta_j^2)}{2} - \sum_{m=S_j+1}^{\infty} (\frac{m-S_j}{2\lambda_j})(m+1-S_j) q_{j,m+1}, \text{ for } j \in J \quad (3.35)$$

The infinite sum is approximated by a finite sum until $q_{j,m+1}$ becomes insignificant. This is the second inaccuracy introduced by the approximation.

Analysis of the second part is quite different from the first part. Let $E(A_j^{2,(k)})$ be the average total inventory in the second region when server faces with k customers. So,

$$E(A_j^{2,(k)}) = \frac{k-1}{\lambda_j+\mu_j} + \frac{\lambda_j}{\lambda_j+\mu_j}E(A_j^{2,(k+1)}) + \frac{\mu_j}{\lambda_j+\mu_j}E(A_j^{2,(k-1)}),$$

When the server faces with k customers, a service will occur before an arrival occurs with probability $\frac{\mu_j}{\lambda_j+\mu_j}$ and the total inventory will be as if there are $k-1$ customers. If an arrival occurs first, the total inventory will be found as if there are $k+1$ customers. Obviously there is inventory of size $(k-1)/(\lambda_j+\mu_j)$ until one of these events occurs. In the light of the above discussion, the following system of linear equations are devised. For $j \in J$,

$$\begin{aligned}
E(A_j^{2,(1)}) &= 0 + \frac{\lambda_j}{\lambda_j+\mu_j}E(A_j^{2,(2)}), \\
E(A_j^{2,(k)}) &= \frac{k-1}{\lambda_j+\mu_j} + \frac{\lambda_j}{\lambda_j+\mu_j}E(A_j^{(k+1)}) + \frac{\mu_j}{\lambda_j+\mu_j}E(A_j^{(k-1)}), k = 2,3,\ldots,S_j-1, \\
E(A_j^{(S_j)}) &= \frac{S_j-1}{\lambda_j+\mu_j} + \frac{\lambda_j}{\lambda_j+\mu_j}(\frac{S_j}{\mu_j} + E(A_j^{2,(S_j)}) + \frac{\mu_j}{\lambda_j+\mu_j}E(A_j^{2,(S_j-1)}).
\end{aligned}$$

Above linear system is a band tridiagonal system, so it will take $O(S_j)$ operations to find $E(A_j^{2,(k)})$'s. Now it is quite easy to find $E(A_j^2)$ as below:

$$E(A_j^2) = \sum_{k=0}^{S_j-1} q_{jk}E(A_j^{2,(k)}) + E(A_j^{2,(S_j)}) \sum_{k=S_j}^{\infty} q_{jk}, \text{ for } j \in J.$$

Finally, the expected total queue content is,

$$E(A_j) = E(A_j^1) + E(A_j^2), \text{ for } j \in J,$$

and the average buffer level is:

$$E(N_j(t)) = \frac{E(A_j)}{E(C)}.$$

3.3 Gated Service Policy

3.3.1 The Exact Model

Gated service policy is the same as exhaustive service policy but new arrivals after the visit instant are left to be processed in the next cycle. Thus, Markov process defined in exhaustive case is not sufficient to model the gated one, because the number of customers left to be served must also be taken into consideration. This is achieved by augmenting the server process $\{Z(t), t \geq 0\}$, where

$$Z(t) = \begin{cases} (1, x_1) & \text{server processing customers in queue } B_1 \\ (2, x_2) & \text{server processing customers in queue } B_2 \\ 3 & \text{server switching from customer 1 to customer 2} \\ 4 & \text{server switching from customer 2 to customer 1} \end{cases}$$

and x_j represents the number of customers left to be processed while server serving queue j and $x_j \in \{1, 2, \ldots, S_j\}$.

Hence, the stochastic process $\{N_j(t), Z(t), t \geq 0, j \in J\}$ is a Markov process with the state space $E \subset E_1 X E_2 X E_Z$. Not all states defined by the cross-product are feasible. For instance states $(1, n_2, (1, 3))$ are not feasible states. Because there may not be 1 customer in queue 1, while the server is procession queue 1 and there is 3 customers left to be processed. So, there are $(S_2 + 1)(\frac{S_1^2+S_1+2}{2}) + (S_1 + 1)(\frac{S_2^2+S_2+2}{2})$ feasible states. We can easily construct the underlying Markov chain with the following state transition rates:

$$\begin{aligned} P(i_1, i_2, x)(i_1 + 1, i_2, x) &= \lambda_1, \quad \text{for } i_1 \in E_1, i_2 \in E_2, x \in E_Z, i_1 \neq S_1, \\ P(i_1, i_2, x)(i_1, i_2 + 1, x) &= \lambda_2, \quad \text{for } i_1 \in E_1, i_2 \in E_2, x \in E_Z, i_2 \neq S_2, \\ P(i_1, i_2, (1, k))(i_1 - 1, i_2, (1, k - 1)) &= \mu_1, \quad \text{for } k > 1, i_1 \in E_1, i_2 \in E_2, \\ P(i_1, i_2, (1, 1))(i_1, i_2, 3) &= \mu_1, \quad \text{for } i_1 \in E_1, i_2 \in E_2, \end{aligned}$$

$$
\begin{aligned}
P(i_1, i_2, (2,k))(i_1, i_2 - 1, (2, k-1)) &= \mu_2, \quad \text{for } k > 1, i_1 \in E_1, i_2 \in E_2, \\
P(i_1, i_2, (2,1))(i_1, i_2, 4) &= \mu_2, \quad \text{for } i_1 \in E_1, i_2 \in E_2, \\
P(i_1, i_2, 3)(i_1, i_2 - 1, (2, i_2)) &= \beta_{12}, \quad \text{for } i_1 \in E_1, i_2 \in E_2, i_2 \geq 1, \\
P(i_1, 0, 3)(i_1, 0, 4) &= \beta_{12}, \quad \text{for } i_1 \in E_1, \\
P(i_1, i_2, 4)(i_1 - 1, i_2, (1, i_1)) &= \beta_{21}, \quad \text{for } i_1 \in E_1, i_2 \in E_2, i_1 \geq 1, \\
P(0, i_2, 4)(0, i_2, 3) &= \beta_{21}, \quad \text{for } i_2 \in E_2,
\end{aligned}
$$

This construction of this infinitesimal generator looks like one in the exhaustive case. There is only one difference. Here the completion of service visit periods are controlled by another variable whereas in the former case it was controlled by queue length. For instance in the fourth equation if there is a departure, server goes for the swithcover although there are other customers in queue 1. But state $(i_1, i_2, (1,1))$ tells us that there is only one customer left to be processes.

After constructing the infinitesimal generator, P, it is trivial to find the steady state distribution of queue length and the average throughput rates. Since all the states communicate, the chain is irreducible and positive recurrent. Therefore, the steady state distribution of the states exists, as in the exhaustive case.

Let, $r_{(i_1,i_2,x)}$ be the steady state probability of state (i_1, i_2, x), and $\mathbf{r}$ be the corresponding vector. So, the system of equations,

$$\mathbf{r}P = 0 \tag{3.36}$$

$$\sum_{r \in E} \mathbf{r} = 1 \tag{3.37}$$

gives the required probabilities. We can get the average throughput rates and the queue length distributions easily, as follows:

$$TH_j = \mu_j \sum_{i_1, i_2, k} r_{(i_1, i_2, (j,k))} \tag{3.38}$$

$$p_{1k} = \sum_{i_2,x} r_{(k,i_2,x)} \tag{3.39}$$

$$p_{2k} = \sum_{i_1,x} r_{(i_1,k,x)} \tag{3.40}$$

If one stores all the elements of the rate matrix P, memory requirements will be $O(|E|^2)$, and the computation time will be about $O(|E|^3)$, to solve the steady state probabilities by Gaussian Elimination. But the rate matrix is quite sparse, approximately three nonzeros per row. So we exploit the sparsity using a sparse solver resulting significant savings in both computation and memory requirements. The use of sparse solver will be explained, in detail, in numerical computation section.

3.3.2 Decomposition

The approach is the same as the one in the exhaustive case. Again the distributions of the vacation times are found. The steps of the previous algorithm is applied to this case.

Vacation Periods

Figure 3.6 shows the phase type representation of η_i if probabilities q_{jk}'s are known.

So, the primary task is to determine the q_{jk}'s. At this time, it is assumed that the stationary probabilities exist. The way to find these probabilities is the same as in the exhaustive case, but in gated service policy these probabilities depend on the cycle time, not on the vacation periods. This is because when server begins to serve a queue in a cycle, the new arrivals while that queue is served and while the server is switching between the queues and serving to another queue should be counted. This duration is called as the cycle time. Therefore q_{jk}s depend on the cycle time distribution. Let C be a random variable representing the cycle time. The following equality holds in general:

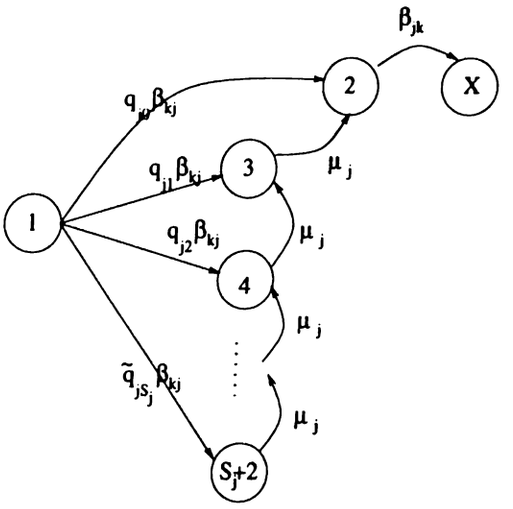

Figure 3.6: Phase-type Representation of $\eta_i, j \neq i$

$$C = \gamma_1 + \xi_{12} + \gamma_2 + \xi_{21} \tag{3.41}$$

If the distribution of C is known, the following system of equations can be obtained.

$$q_{jk} = \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!) f_C(u) du, k = 0, 1, ..N_j - 1, j = 1, 2$$

$$\tilde{q}_{jS_j} = \sum_{k=S_j}^{\infty} \int_0^\infty (e^{-\lambda_j u}(\lambda_j u)^k/k!) f_C(u) du, j = 1, 2 \tag{3.42}$$

As shown in Figure 3.7, C has a phase-type representation. Cycle time is composed of two switchover times and two visit periods. States $S_2 + 1$ and $S_2 + S_1 + 2$ are the states representing the switchovers. Likewise states 1 through S_2 belong to the visit period of the second queue and states $S_2 + 2$ through $S_2 + S_1 + 1$ belong to the visit period of the first queue Let Q be the infinitesimal generator of the cycle time. Then, it is obvious from the previous section that,

$$Q = \begin{bmatrix} A & A^0 \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.43}$$

For the whole range of parameters, the absorption into state **X** is guaranteed. Now one can determine A and the initial probability vector α.

The matrix A is:

$$A = \begin{bmatrix} -\mu_2 & \mu_2 & 0 & & 0 & 0 & 0 & & 0 & 0 \\ 0 & -\mu_2 & \mu_2 & \cdots & 0 & 0 & 0 & & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\beta_{21} & \tilde{q}_{1,S_1}\beta_{21} & q_{1,S_1-1}\beta_{21} & \cdots & q_{11}\beta_{21} & q_{10}\beta_{21} \\ 0 & 0 & 0 & & 0 & -\mu_1 & \mu_1 & & 0 & 0 \\ 0 & 0 & 0 & & 0 & 0 & -\mu_1 & & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & & 0 & 0 & 0 & & -\mu_1 & \mu_1 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & & 0 & \beta_{12} \end{bmatrix},,$$

and, the initial probability vector,

$$\alpha = (\tilde{q}_{2,S_2}, q_{2,S_j-1}, \ldots, q_{21}, q_{20}, 0, \ldots, 0, 0)$$

But, it is obvious that Equations (3.42) do not reduce to a system of linear equations. Newton's method could be used to solve the system of nonlinear equations. But again extensive numerical calculations are needed to find the parameters of the equations. Therefore, a similar fixed point algorithm is proposed to solve the system. But neither the convergence of the algorithm nor existence of unique stationary probabilities q_{jk}s could be proven.

Although matrix A is not a band tridiagonal one, it is a singly bordered band diagonal matrix. So Gaussian elimination would take a time that is linear with the size of the matrix.

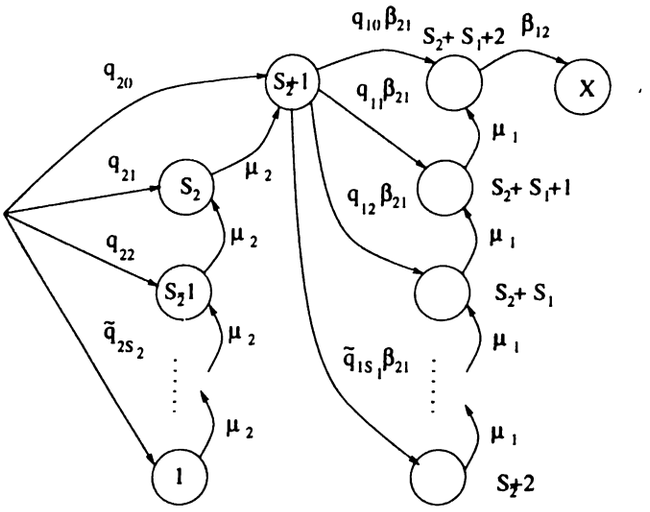

Figure 3.7: Phase-type Representation of Cycle Time C

Decomposed Models

Let the stochastic process $\{Z_j(t), t \geq 0\}, j \in J$ representing the state of the pseudo-server j where $Z_j(t) \in E_{Z_j} = \{1, 2, \ldots, S_1 + S_2 + 2\}$,such that

$$Z_j(t) = \begin{cases} 1 & \text{pseudo-server } j \text{ is serving and 1 customer left to be served} \\ 2 & \text{pseudo-server } j \text{ is serving and 2 customers left to be served} \\ \vdots & \\ S_j & \text{pseudo-server } j \text{ serving and } S_j \text{ customers left to be served} \\ S_j + 1 & \text{pseudo-server is at vacation in stage 1} \\ S_j + 2 & \text{pseudo-server is at vacation in stage 2} \\ \vdots & \\ S_j + (S_i + 2) & \text{pseudo-server is at vacation in stage } (S_i + 2) \end{cases}$$

where $i \neq j$.

So, one server systems can be derived as the stochastic processes $\{N_j(t), Z_j(t), t \geq 0\}, j \in J$. Since the vacation periods are of phase type, these processes are Markov processes with the state spaces $E'_j \subset E_j X E_{Z_j}$ where $|E'_1| = \frac{S_1(S_1+3)}{2} + (S_1 + 1)(S_2 + 2)$ and $|E'_2| = \frac{S_2(S_2+3)}{2} + (S_2 + 1)(S_1 + 2)$.

Markov Chains of the above Markov processes can be easily constructed. Figure 3.6 can be helpful to understand the following equations.

$$
\begin{aligned}
P_1(i,k)(i+1,k) &= \lambda_1, && \text{for } i \neq S_1, k \in E_{Z_1},\\
P_1(i,k)(i-1,k-1) &= \mu_1, && \text{for } i \in E_1, k > 1,\\
P_1(i,1)(i,S_1+1) &= \mu_1, && \text{for } i \in E_1,\\
P_1(i,S_1+1)(i,k) &= q_{2,(k-S_1-2)}\beta_{12}, && \text{for } i \in E_1,\\
& && (S_1+2) \leq k \leq (S_1+S_2+1),\\
P_1(i,S_1+1)(i,S_1+S_2+2) &= \tilde{q}_{2,S_2}\beta_{12}. && \text{for } i \in E_1,\\
P_1(i,k)(i,k-1) &= \mu_2, && \text{for } i \in E_1,\\
& && (S_1+3) \leq k \leq (S_1+S_2+2),\\
P_1(i,S_1+S_2+2)(i-1,i) &= \beta_{21}, && \text{for } i \in E_1, i > 0,\\
P_1(0,S_1+S_2+2)(0,S_1+1) &= \beta_{21}, &&
\end{aligned}
$$

Like the exact case, decomposed model also looks like the one in exhaustive service policy. The first equation states the arrival rate of the customer. In the second one there are departures from the system with rate μ_1. The third one states that if there is one customer left to be processed, server takes a vacation with rate μ_1. The other states deals with the vacation. But here, unlike the exhaustive case, arrivals to second customers are not considered. And the last equation states that, if there is no customers waiting in queue 1, server takes another vacation. Construction of the second generator easily follows:

$$
\begin{aligned}
P_2(i,k)(i+1,k) &= \lambda_2, && \text{for } i \neq S_2, k \in E_{Z_2},\\
P_2(i,k)(i-1,k-1) &= \mu_2, && \text{for } i \in E_2, k > 1,\\
P_2(i,1)(i,S_2+1) &= \mu_2, && \text{for } i \in E_2,\\
P_2(i,S_2+1)(i,k) &= q_{2,(k-S_2-2)}\beta_{21}, && \text{for } i \in E_2,\\
& && (S_2+2) \leq k \leq (S_2+S_1+1),\\
P_2(i,S_2+1)(i,S_2+S_1+2) &= \tilde{q}_{1,S_1}\beta_{21}, && \text{for } i \in E_2,\\
P_2(i,k)(i,k-1) &= \mu_2, && \text{for } i \in E_2,\\
& && (S_2+3) \leq k \leq (S_2+S_1+2),\\
P_2(i,S_2+S_1+2)(i-1,i) &= \beta_{12}, && \text{for } i \in E_2, i > 0,\\
P_2(0,S_2+S_1+2)(0,S_2+1) &= \beta_{12}, &&
\end{aligned}
$$

After constructing the infinitesimal generators, $P_j, j \in J$, it is trivial to find the steady state distribution of queue length and the average throughput rates, since all the states communicate, the chains are irreducible and recurrent.

Let $r_j(i,k)$ be the steady state probability of state (i,k) and $\mathbf{r_j}$ be the corresponding vectors. So, system of equations,

$$\mathbf{r_j} P_j = 0,$$

$$\sum_{\mathbf{r_j} \in E'_j} \mathbf{r_j} = 1, \qquad \text{for } j \in J, \tag{3.44}$$

gives the steady state probabilities. We can obtain the average throughput rates and the queue length distributions easily, as follows:

$$TH_j = \mu_j \sum_{i,k \leq S_j} r_j(i,k) \tag{3.45}$$

$$p_{jm} = \sum_{k \in E_{Z_j}} r_j(m,k) \tag{3.46}$$

3.3.3 Approximation

Stochastic processes $\{Z(t), t \geq 0\}$ and $\{N_j(t), t \geq 0\}, j \in J$ are *regenerative processes*, with the state spaces $\{1,2,3,4\}$, and E_j respectively because, at every visit beginnings of a particular queue, the process restarts itself probabilistically. In this section, the cycle time is defined as the time between successive visit beginnings. It is equivalent to the previous definition. The definitions of thruput rates are the same as in the exhaustive case. But the analysis of the average inventory level is somewhat different. Figure 3.8 shows a particular realization of the process $\{N_j(t), t \geq 0\}$ in a cycle. Here the cycle time is defined as the time between successive visit beginnings to a queue. The first part shows the process during the visit period. The second part is the inventory accumulation during the vacation period.

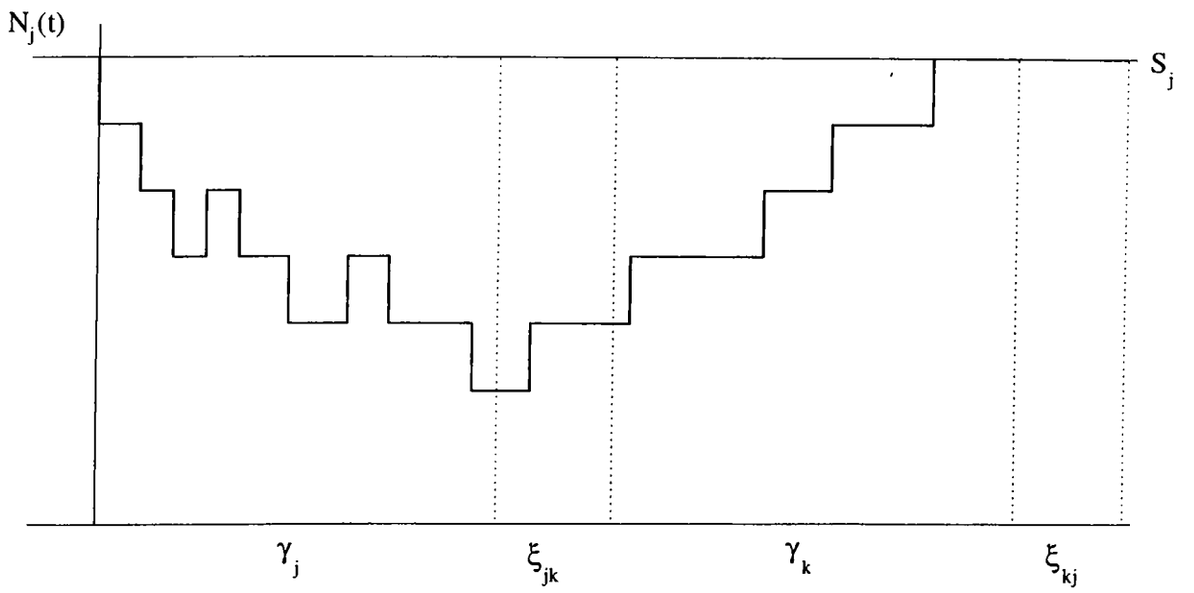

Figure 3.8: Gated Case: A Particular Realization of N_j, $j \neq k$.

In this case the inventory process in a cycle is decomposed differently. As it can be seen in Figure 3.9, the inventory process is decomposed into two inventory processes, $N_j^1(t)$, and $N_j^2(t)$, which can be characterized by a departure process and an arrival process. Let, A_j be the area under the process $\{N_j(t), t \geq 0\}$, in a cycle. Then,

$$E[N_j(t)] = E[A_j]/E(C) \tag{3.47}$$

where,

$$A_j = A_j^1 + A_j^2, \text{ for } j \in J. \tag{3.48}$$

Let $\{s_k, k \geq 0\}$ and $\{t_k, k \geq 0\}$ be the departure and arrival points during the cycle time. Suppose that server faces with m customers when he visit the queue j, then it is obvious that,

$$A_j^1 = \sum_{n=1}^{m} (s_n - s_{n-1})(m - n), \text{ for } j \in J \tag{3.49}$$

therefore

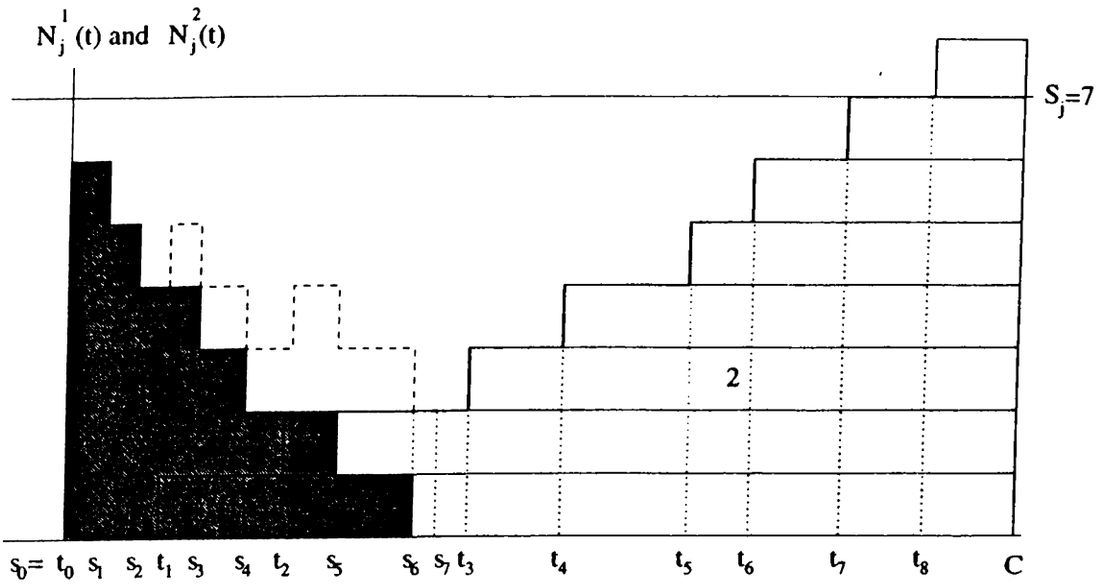

Figure 3.9: Gated Case:Total Inventory Process During the Cycle Time

$$E(A_j^1|N(C)=m)=\sum_{n=1}^{m}\frac{(n-1)}{\mu_j}=\frac{(m-1)m}{2\mu_j},\ \text{for } j\in J,\ \text{and } m=1,2,\ldots,S_j, \tag{3.50}$$

then,

$$E(A_j^1)=\sum_{m=1}^{S_j-1}\frac{(m-1)m}{2\mu_j}q_{jm}+\frac{(S_j-1)S_j}{2\mu_j}\sum_{m=S_j}^{\infty}q_{jm},\ \text{for } j\in J, \tag{3.51}$$

The analysis of the second part is exactly the same as in the exhaustive case, as long as it is assumed that there is no lost customers during the visit period. Here, the vacation period is replaced by the cycle time. Therefore,

$$E(A_j^2)=\frac{\lambda_j E(C^2)}{2}-\sum_{m=S_j+1}^{\infty}(\frac{m-S_j}{2\lambda_j})(m+1-S_j)q_{j,m+1},\ \text{for } j\in J \tag{3.52}$$

Hence, the mean queue length is,

$$E[N_j(t)]=\frac{E(A_j^1)+E(A_j^2)}{E(C)},\ \text{for } j\in J \tag{3.53}$$

Notice that the approximation accuracy decreases if the proportion of the lost customers during the visit period increases.

3.4 G-Limited Service Policy

In the literature k-limited policy is referred as serving in fixed, and pre-determined batches. But this definition is not sufficient to determine the policy exactly. A class of alternative policies can be deduced. If server finds more than k number of customers, it serves k of them and switches to the other queue. But if it cannot find, as much as k customers, it can choose one among the following three policies: It waits until it serves k customers, he serves only the customers which are waiting for service (like gated service policy) and it switches when it *clears* the queue unless the number of customers exceed k. The second policy is referred as G-limited in the literature.

Therefore G-limited policy can be summarized as follows: If server faces with as much as k customers, it serves k of them in that cycle, otherwise it serves the customers that exist at the visiting instant, as in the case of gated service policy.

3.4.1 The Exact Model

In this part, the exact Markov model is constructed. Augmenting the server process $\{Z(t), t \geq 0\}$ here is inevitable also. It is done as follows:

$$Z(t) = \begin{cases} (1, x_1) & \text{server processing customers in queue } B_1 \\ (2, x_2) & \text{server processing customers in queue } B_2 \\ 3 & \text{server switching from customer 1 to customer 2} \\ 4 & \text{server switching from customer 2 to customer 1} \end{cases}$$

and x_j represents the number of customers left to be processed while server processing queue j. So, $x_j \in \{1, 2, \ldots, K_j\}$, where K_j is predetermined queue-dependent batch sizes.

Hence, the stochastic process $\{N_j(t), Z(t), t \geq 0, j \in J\}$ is a Markov process with the state space $E \subset E_1 X E_2 X E_Z$, where $E_Z = \{(1,1),(1,2),\ldots,(1,K_1),(2,1),(2,2),\ldots,(1,K_2),3,4\}$. Not all the states defined by the cross-product are feasible. For instance the states $(1,n_2,(1,3))$ are not feasible states. So, there are $2(S_1+1)(S_2+1)+(S_1+1)(\sum_{m=1}^{K_2}\sum_{n=m}^{S_2} n)+(S_2+1)(\sum_{m=1}^{K_1}\sum_{n=m}^{S_1} n)$ feasible states. We can easily construct the underlying Markov chain with the following state transition rates:

$$
\begin{aligned}
P(i_1,i_2,x)(i_1+1,i_2,x) &= \lambda_1, && \text{for } i_1 \in E_1, i_2 \in E_2, x \in E_Z, i_1 \neq S_1,\\
P(i_1,i_2,x)(i_1,i_2+1,x) &= \lambda_2, && \text{for } i_1 \in E_1, i_2 \in E_2, x \in E_Z, i_2 \neq S_2,\\
P(i_1,i_2,(1,k))(i_1-1,i_2,(1,k-1)) &= \mu_1, && \text{for } k>1, i_1 \in E_1, i_2 \in E_2,\\
P(i_1,i_2,(1,1))(i_1,i_2,3) &= \mu_1, && \text{for } i_1 \in E_1, i_2 \in E_2,\\
P(i_1,i_2,(2,k))(i_1,i_2-1,(2,k-1)) &= \mu_2, && \text{for } k>1, i_1 \in E_1, i_2 \in E_2,\\
P(i_1,i_2,(2,1))(i_1,i_2,4) &= \mu_2, && \text{for } i_1 \in E_1, i_2 \in E_2,\\
P(i_1,i_2,3)(i_1,i_2-1,(2,i_2)) &= \beta_{12}, && \text{for } i_1 \in E_1, i_2 \in E_2, 1 \leq i_2 \leq K_2,\\
P(i_1,i_2,3)(i_1,i_2-1,(2,K_2)) &= \beta_{12}, && \text{for } i_1 \in E_1, i_2 \in E_2, K_2 < i_2,\\
P(i_1,0,3)(i_1,0,4) &= \beta_{12}, && \text{for } i_1 \in E_1,\\
P(i_1,i_2,4)(i_1-1,i_2,(1,i_1)) &= \beta_{21}, && \text{for } i_1 \in E_1, i_2 \in E_2, 1 \leq i_1 \leq K_1,\\
P(i_1,i_2,4)(i_1-1,i_2,(1,K_1)) &= \beta_{21}, && \text{for } i_1 \in E_1, i_2 \in E_2, K_1 \leq i_1,\\
P(0,i_2,4)(0,i_2,3) &= \beta_{21}, && \text{for } i_2 \in E_2,
\end{aligned}
$$

The first two equations state that new arrivals join their queues if they are not full. The remaining equations are the same with the ones seen in gated case but, here S_j is replaced by K_j.

After constructing the infinitesimal generator,P it is trivial to find the steady state distribution of the queue length and the average throughput rates. Since all the states communicate, the chain is irreducible and recurrent. So, the steady state distribution exists. Calculation of the performance measures are exactly the same as in the gated service policy case.

3.4.2 Decomposition

Vacation Periods

Let q_{jm} be the probability of being m customers in queue B_j, when the server is ready for the processing customer j. These are again the key probabilities to find the distributions of vacation periods and the cycle times.

So, the primary task is to determine the q_{jm}'s. The way to find these probabilities is the same as of gated policy. But in G-limited service policy these probabilities depend not only on the cycle time distribution but also the queue length in the previous cycle. New variables are defined to simplify the expressions:

B_{jm}^l: the event that the server faces with m customers whenever he turns for queue j in cycle l,

C_{jm}^l: the event that there will be m new arrivals to queue j during cycle time in cycle l,

D_{jm}^l: the event that there will be m new arrivals to queue j during the vacation period of queue j, in cycle l.

So, the following equations define q_{jm}^l:

$$q_{jm}^l = P(B_{jm}^l) \qquad m = 0, 1, \ldots, S_j, \qquad j = 1, 2, \tag{3.54}$$

By the total probability law:

$$q_{jm}^l = \sum_{m'=0}^{S_j} P(B_m^l | B_{m'}^{l-1}) q_{jm'}^{l-1}, \; m = 0, 1, \ldots, S_j, \; j = 1, 2 \tag{3.55}$$

which are equivalent to:

$$q_{jm}^l = q_{j0}^{l-1} P(D_{jm}^l) + \sum_{m'=1}^{K_j} q_{jm'}^{l-1} P(C_{j,m}^l) + \sum_{m'=k+1}^{S_j} q_{jm'}^{l-1} P(C_{j,(m-m'+k)}^l), \tag{3.56}$$
$$m = 0, 1, \ldots, S_j - 1, \; j = 1, 2$$

Assume that $q_{jk}^l \to q_{jk}$ as $l \to \infty$. So, above equation becomes:

$$q_{jm} = q_{j0}P(D_{jm}) + \sum_{m'=1}^{K_j} q_{jm'}P(C_{j,m}) + \sum_{m'=k+1}^{S_j} q_{jm'}P(C_{j,(m-m'+k)}),$$
$$m = 0, 1, \ldots, S_j - 1, \; j = 1, 2 \quad (3.57)$$

If there are the cycle time and the vacation period distributions, B_{jm} and C_{jm} are found as follows:

$$C_{jm} = \int_0^\infty e^{-\lambda_j u}(\lambda_j u)^m/m! f_C(u)du \qquad m = 0, 1, \ldots, S_j. \quad j = 1, 2 \quad (3.58)$$

$$D_{jm} = \int_0^\infty e^{-\lambda_j u}(\lambda_j u)^m/m! f_{\eta_j}(u)du \qquad m = 0, 1, \ldots, S_j. \quad j = 1, 2 \quad (3.59)$$

And finally, the boundary values of q_{jm} are:

$$\tilde{q}_{jS_j} = 1 - \sum_{k=1}^{S_j - 1} q_{jk}, \qquad j = 1, 2, \text{ and,}$$

$$\tilde{q}_{jK_j} = 1 - \sum_{k=1}^{K_j - 1} q_{jk}, \qquad j = 1, 2, \text{ and,}$$

Figure 3.10 shows the phase type representation of the clearance times. Vacation periods can be obtained by adding two switchovers as shown in Figure 3.11.

Now let us determine the infinitesimal generator, Q_j, and the initial probability vector α_j of the vacation periods. It is obvious that Q_j has the form of,

$$Q_j = \begin{bmatrix} A_j & A_j^0 \\ \mathbf{0} & 0 \end{bmatrix} \quad (3.60)$$

where the $(K_i + 2)$ X $(K_i + 2)$ matrix $A_j, i \neq j$, can be written as:

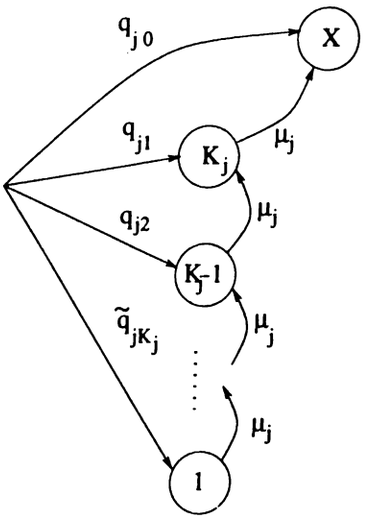

Figure 3.10: Phase-type Representation of γ_j, $\tilde{q}_{jK_j} = \sum_{m=K_j}^{S_j} q_{jm}$

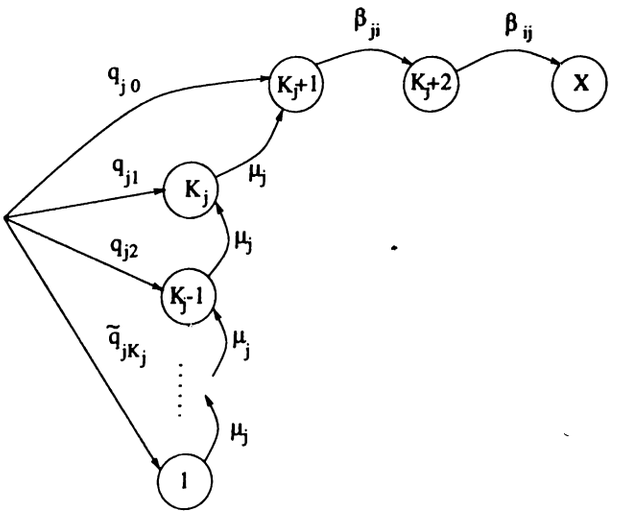

Figure 3.11: Phase-type Representation of η_i, $\tilde{q}_{jK_j} = \sum_{m=K_j}^{S_j} q_{jm}, i \neq j$

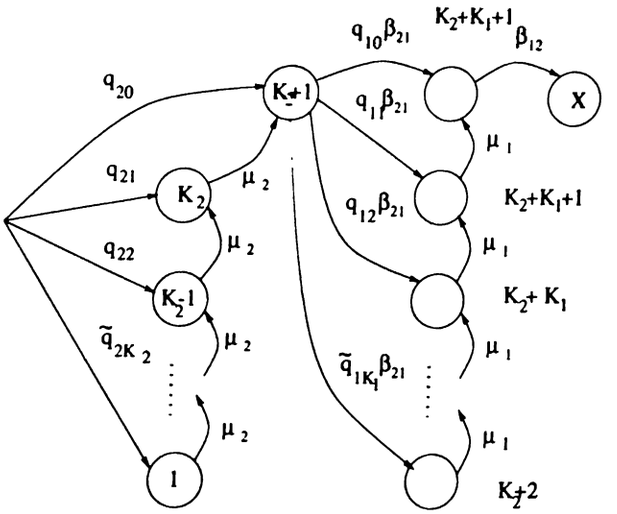

Figure 3.12: Phase-type Representation of C, $\tilde{q}_{jK_j} = \sum_{m=K_j}^{S_j} q_{jm}, j = 1, 2.$

$$A_j = \begin{bmatrix} -\mu_i & \mu_i & 0 & & 0 & 0 & 0 \\ 0 & -\mu_i & \mu_i & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -\mu_i & \mu_i & 0 \\ 0 & 0 & 0 & & 0 & -\beta_{ji} & \beta_{ji} \\ 0 & 0 & 0 & \dots & 0 & 0 & -\beta_{ij} \end{bmatrix},$$

and, $A_j\mathbf{e} + A_j^0 = \mathbf{0}$. The initial probability vector α_j is

$$\alpha_\mathbf{j} = (\tilde{q}_{i,K_i}, q_{i,K_i-1}, \dots, q_{i1}, q_{i0}, 0).$$

Here all the states except the state **X** are transient, that is absorption into the state **X** is certain. For the whole range of parameters this condition is guaranteed. Let us now determine the infinitesimal generator, Q and the initial probability vector α of the cycle time. for which the phase-type representation is given in Figure 3.12. Q is similar to the one in the gated case. Let,

$$Q = \begin{bmatrix} A & A^0 \\ \mathbf{0} & 0 \end{bmatrix} \tag{3.61}$$

where, A is a $(K_1 + K_2 + 2)$ X $(K_1 + K_2 + 2)$ matrix with the following structure:

$$A = \begin{bmatrix} -\mu_2 & \mu_2 & 0 & & 0 & 0 & 0 & & 0 & 0 \\ 0 & -\mu_2 & \mu_2 & \cdots & 0 & 0 & 0 & & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\beta_{21} & \tilde{q}_{1,K_1}\beta_{21} & q_{1,K_1-1}\beta_{21} & \cdots & q_{11}\beta_{21} & q_{10}\beta_{21} \\ 0 & 0 & 0 & & 0 & -\mu_1 & \mu_1 & & 0 & 0 \\ 0 & 0 & 0 & & 0 & 0 & -\mu_1 & & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & -\mu_1 & \mu_1 \\ 0 & 0 & 0 & & 0 & 0 & 0 & & 0 & \beta_{12} \end{bmatrix},$$

and, the initial probability vector is:

$$\alpha = (\tilde{q}_{2,K_2}, q_{2,K_2-1}, \ldots, q_{21}, q_{20}, 0, \ldots, 0, 0)$$

But the system of equations (3.57) is a nonlinear one. A similar iterative algorithm can be proposed as in the gated policy. Again, neither convergence of the algorithm nor uniqueness of these probabilities could be proven.

The Model

Let the stochastic process $\{Z_j(t), t \geq 0\}, j \in J$ representing the state of the pseudo-server j where $Z_j(t) \in E_{Z_j} = \{1, 2, \ldots, K_1 + K_2 + 2\}$,such that

$$Z_j(t) = \begin{cases} 1 & \text{pseudo-server } j \text{ is serving and 1 customer left to be served} \\ 2 & \text{pseudo-server } j \text{ is serving and 2 customers left to be served} \\ \vdots & \\ K_j & \text{pseudo-server } j \text{ serving and } K_j \text{ customers left to be served} \\ K_j+1 & \text{pseudo-server is at vacation in stage 1} \\ K_j+2 & \text{pseudo-server is at vacation in stage 2} \\ \vdots & \\ K_j+(K_i+2) & \text{pseudo-server is at vacation in stage } (K_i+2) \end{cases}$$

Again we can model the one server models as the stochastic processes $\{N_j(t), Z_j(t), t \geq 0\}, j \in J$. Since the vacation periods are of phase type these processes are Markov processes with the state spaces $E'_j \subset E_j X E_{Z_j}$ where $|E'_1| = (K_1+1)(K_2+2) + (\sum_{m=1}^{K_1}\sum_{n=m}^{K_1} n)$, and $|E'_2| = (K_2+1)(K_1+2) + (\sum_{m=1}^{K_2}\sum_{n=m}^{K_2} n)$.

Markov Chains of the above Markov processes can be easily constructed as follows:

$$\begin{aligned}
P_1(i,k)(i+1,k) &= \lambda_1, && \text{for } i \neq K_1, k \in E_{Z_1}, \\
P_1(i,k)(i-1,k-1) &= \mu_1, && \text{for } i \in E_1, 1 < k \leq K_1, \\
P_1(i,1)(i,K_1+1) &= \mu_1, && \text{for } i \in E_1, \\
P_1(i,K_1+1)(i,k) &= q_{2,(k-K_1-2)}\beta_{12}, && \text{for } i \in E_1, \\
& && (K_1+2) \leq k < (K_1+K_2+2), \\
P_1(i,K_1+1)(i,K_1+K_2+2) &= \tilde{q}_{2,K_2}\beta_{12}, && \text{for } i \in E_1, \\
P_1(i,k)(i,k-1) &= \mu_2, && \text{for } i \in E_1, \\
& && (K_1+3) \leq k \leq (K_1+K_2+2), \\
P_1(i,K_1+K_2+2)(i-1,i) &= \beta_{21}, && \text{for } i \in E_1, i > 0, \\
P_1(0,K_1+K_2+2)(0,K_1+1) &= \beta_{21}, &&
\end{aligned}$$

The first equation states that new arrivals join queue 1 if it is not full. The remaining equations are the same with the ones seen in gated case but, here

S_j is replaced by K_j. Construction of the second sub-system easily follows:

$$\begin{aligned}
P_2(i,k)(i+1,k) &= \lambda_2, && \text{for } i \neq K_2, k \in E_{Z_2},\\
P_2(i,k)(i-1,k-1) &= \mu_2, && \text{for } i \in E_2, 1 < k \leq K_2,\\
P_2(i,1)(i,K_2+1) &= \mu_2, && \text{for } i \in E_2,\\
P_2(i,K_2+1)(i,k) &= q_{2,(k-K_2-2)}\beta_{21}, && \text{for } i \in E_2,\\
& && (K_2+2) \leq k < (K_2+K_1+2),\\
P_2(i,K_2+1)(i,K_1+K_2+2) &= \tilde{q}_{2,K_2}\beta_{21}, && \text{for } i \in E_2,\\
P_2(i,k)(i,k-1) &= \mu_2, && \text{for } i \in E_2,\\
& && (K_2+3) \leq k \leq (K_2+K_1+2),\\
P_2(i,K_2+K_1+2)(i-1,i) &= \beta_{12}, && \text{for } i \in E_2, i > 0,\\
P_2(0,K_2+K_1+2)(0,K_2+1) &= \beta_{12},
\end{aligned}$$

After constructing the infinitesimal generators,$P_j, j \in J$ it is trivial to find the steady state distribution of the queue length and the average throughput rates. Since all the states communicate, the chains are irreducible and recurrent. Therefore, the steady state probabilities exist. Let $r_j(i,k)$ be the steady state probability of state (i,k) and $\mathbf{r_j}$ be the corresponding vectors. So, system of equations,

$$\mathbf{r_j} P_j = 0,$$

$$\sum_{\mathbf{r_j} \in E'_j} \mathbf{r_j} = 1, \qquad \text{for } j \in J, \tag{3.62}$$

gives the steady state probabilities. We can obtain the average throughput rates and the queue length distributions easily, as follows:

$$TH_j = \mu_j \sum_{i,k \leq K_j} r_j(i,k) \tag{3.63}$$

$$p_{jm} = \sum_{k \in E_{Z_j}} r_j(m,k), \tag{3.64}$$

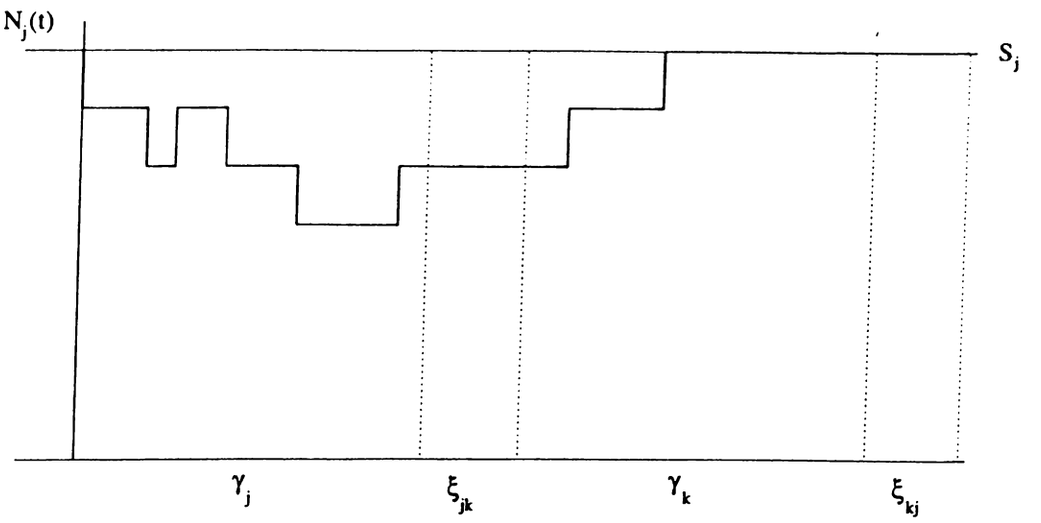

Figure 3.13: G-Limited Case:A Particular Realization of N_j, $j \neq k$.

3.4.3 Approximation

The stochastic processes $\{Z(t), t \geq 0\}$ and $\{N_j(t), t \geq 0\}, j \in J$ are *regenerative processes*, with state space $\{1, 2, 3, 4\}$, and E_j respectively, because at every visit beginnings of a particular queue, the process restarts itself probabilistically. In this section the cycle time is defined as the time between successive visit beginnings. The definitions of thruput rates are the same as in the exhaustive case. But, the analysis of the average inventory level is somewhat different. Figure 3.13 shows a particular realization of the process $\{N_j(t), t \geq 0\}$ in a cycle. Although this case is different from the previous cases, similarities with the gated case can be observed. But, here the inventory process is decomposed into three processes, where N_j^1 represents the inventory process due to departures, N_j^2 is the inventory process due to arrivals. N_j^3 is determined by the number of customers that server faces with at the visit instant.

Figure 3.14 will be helpful to understand this decomposition. Let A_j be the area under the process $\{N_j(t), t \geq 0\}$, in a cycle. Then,

$$E[N_j(t)] = E[A_j]/E(C), \text{ where} \tag{3.65}$$

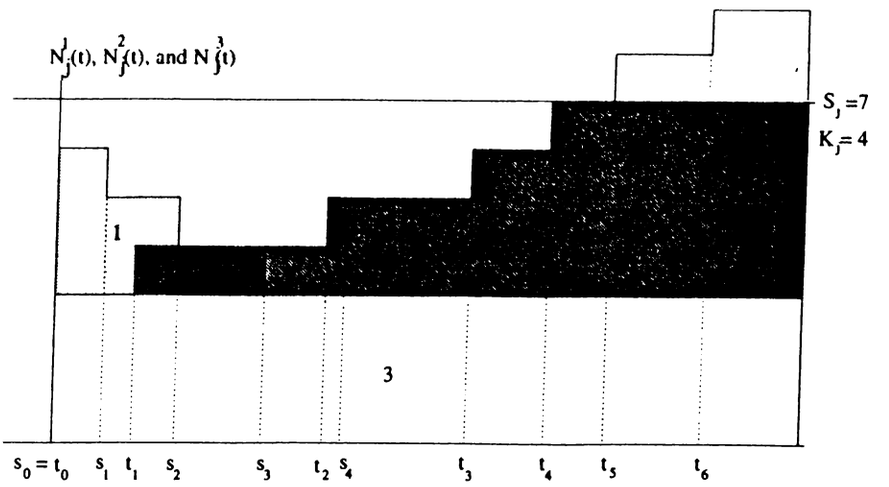

Figure 3.14: G-Limited Case:Total Inventory Process During the Cycle Time

$$A_j = A_j^1 + A_j^2 + A_j^3, \text{ for } j \in J. \tag{3.66}$$

It is clear from the previous section that,

$$E(A_j^1) = \sum_{m=1}^{K_j-1} \frac{(m-1)m}{2\mu_j} q_{jm} + \frac{(K_j-1)K_j}{2\mu_j} \sum_{m=K_j}^{S_j} q_{jm}, \text{ for } j \in J, \text{ and,} \tag{3.67}$$

$$E(A_j^3) = \sum_{m=K_j+1}^{S_j} (m-K_j)E(C). \tag{3.68}$$

Analysis is of the second part is slightly different. If the server faces with less than or equal to K_j customers, it is the same as in the gated case with buffer size S_j. But if it faces with more than K_j customers, it resembles the gated policy with buffer size of $S_j - n + K_j$. Therefore,

$$E(A_j^2|B_{jn}, n \leq K_j) = \frac{\lambda_j E(C^2)}{2} - \sum_{m=S_j+1}^{\infty} (\frac{m-S_j}{2\lambda_j})(m+1-S_j)r_{j,m+1}, \text{ for } j \in J, \text{ and}$$

$$E(A_j^2|B_{jn}, n \geq K_j) = \frac{\lambda_j E(C^2)}{2} - \sum_{m=S_j^n+1}^{\infty} (\frac{m-S_j^n}{2\lambda_j})(m+1-S_j^n)r_{j,m+1}, \text{ for } j \in J, \text{ and}$$

where $S_j^n = S_j - n + K_j$, and $r_{j,m} = P\{m \text{ arrivals during the cycle time}\} = P\{B_{jm}\}$. Hence,

$$E(A_j^2) = \sum_{n=1}^{K_j} E(A_j^2|B_{jn}, n \leq K_j)q_{jn} + \sum_{n=K_j}^{S_j} E(A_j^2|B_{jn}, n \geq K_j)q_{jn}, j \in J. \quad (3.69)$$

Chapter 4

Numerical Results

In the previous chapter, analytical results belonging to the different service policies are given. For each service policy three solution methods are proposed: exact, decomposition and approximation. In this chapter, the numerical results concerning these solution methods will be discussed for each service policy. These results are divided into two main parts: Computational efforts required and accuracy of the decomposition and approximation in the estimation of the performance measures. The performance measures are steady state thruput rates and average buffer levels. To report these results meaningfully, sets of problems, generated through an experimental design strategy, are solved. There are three design parameters: Buffer sizes, switchovers, and loads, or traffic intensities. Load is defined as the proportion of arrival rate to processing rate. Batch sizes are other design parameters in the G-limited case.

The organization of the chapter is as follows: next section explains the coding environment of the methods. Succeeding sections are devoted to the results of the different service policies.

4.1 Computer Codes

As it is seen from the previous chapter three solution methods are proposed for each service policy. To code these methods MATLAB script files and C programs are used. In this section rough steps of the codes will be given.

In the exact method, the construction of the infinitesimal generator of the Markov process, the computation of the steady state probabilities using this generator and finally the calculation of the steady state distribution of states and performance measures are the main steps. First two steps are coded in MATLAB, in this phase of the algorithm all feasible states are generated and infinitesimal generator is written to a file in a sparse format. Next, this file used as an input to a sparse solver and solution is obtained via a text file. This file is mainly used as an input to MATLAB and the performance measures are calculated there. In summary:

Algorithm 4.1 *(Exact)*
1.Generation of the states of the Markov process, (MATLAB),
2.Construction of the infinitesimal generator, (MATLAB),
3.Calculation of steady state probabilities, (Sparse Solver),
4.Calculation of performance measures, (MATLAB).

While finding the computational requirements of the exact method, only the third step is considered, that is the calculation of the steady state distribution of the states. Because the main numerical operations are performed in this step. The other steps are about the construction, some search algorithms and the calculation of the performance measures.

Algorithm of the decomposition is very similar to the exact one. But there are two Markov processes and, additionally, facing probabilities (q_{jk}'s) should be calculated. In summary:

Algorithm 4.2 *(Decomposition)*
1.Calculation of facing probabilities, (C),
2.Determination of the states of the Markov processes, (MATLAB),
3.Construction of the infinitesimal generators, (MATLAB),
4.Calculation of steady state probabilities (Sparse Solver),
5.Calculation of performance measures (MATLAB).

The first step is performed with a C program, which is the iterative algorithm explained previously. The computational efforts of the decomposed models are found by summing the times required via step one and step four.

Algorithm of the approximation is a very simple one:

Algorithm 4.3 *(Approximation)*
1.Calculation of extended facing probabilities, (C),
2.Calculation of performance measures (MATLAB).

In Step 1 we use the term *extended*, because the usual facing probabilities and more are required, as explained previously.

As one notices the computation time requirements of the methods are the steps coded in C. MATLAB is used only for construction purposes and calculation of performance measures. Also, the computations are conducted on a SUN-WorkStation/4.25 with 64 MB memory. The fixed point algorithm is run until the difference between two successive probabilities is less than a fixed ϵ. In all experiments $\epsilon = 10^{-6}$, with a maximum allowable fixed point iteration of 99. All facing probabilities are initialized to uniform values as it is stated in Algorithm 3.1.

For solving systems of linear equations a sparse solver called Sparse1.3, written by K.S. Kundert and A.S. Vincentelli from Department of Electrical Engineering and Computer Sciences in University of California, Berkeley, is

used. It is is a flexible package of subroutines written in C used to quickly and accurately solve large sparse systems of linear equations. Sparse1.3 is generally as fast as or faster than other popular sparse matrix packages when solving many matrices of similar structure. They compared Sparse1.3 to Sparse1.2, Harwell's MA28, and Yale's YSMP for several famous test problems. In most of the test problems Sparse1.3 outperformed other packages [18]. These subroutines are available via NetLib.

4.2 Exhaustive Case

To test the computational savings and the accuracy in the estimation of performance measures between exact model and decomposition and approximation, 6x2x12=144 problems were generated and solved. Generation of the problems are accomplished using the experimental design strategy explained previously. Table 4.1 gives the parameters of the problems.

1.Buffer Sizes	$S_1 = S_2$
	1,2,5,10,15,20
2.Switchovers	β_{ij}
Short	μ_i
Long	$\mu_i/10$
3.Loads	$(\lambda_1, \mu_1, \lambda_2, \mu_2)$
High and High	(0.8,1.0,0.8,1.0), (0.7,1.0,1.4,2.0)
Medium and High	(0.5,1.0,0.8,1.0), (0.4,1.0,1.4,2.0)
Light and High	(0.2,1.0,0.8,1.0), (0.1,1.0,1.4,2.0)
Medium and Medium	(0.5,1.0,0.5,1.0), (0.4,1.0,0.8,2.0)
Light and Medium	(0.2,1.0,0.5,1.0), (0.1,1.0,0.8,2.0)
Light and Light	(0.2,1.0,0.2,1.0), (0.1,1.0,0.2,2.0)

Table 4.1: Problem generation parameters for the exhaustive case

As it is seen the maximum buffer size, solved, is 20. This is because of the huge amount of computational requirements in the construction phase. Nevertheless, this maximum size is enough for most purposes, because as it increases

the model will be more likely to be an infinite queue model for which more simple solution methods exist in the literature [7]. The load conditions are labeled as high, medium and light. In the high load conditions, it is aimed that more than half of the customers should be lost. In the light load conditions it is aimed that there is no lost customers. Note that there are two sets of problems for similar load conditions. In the first set, the processing times of each customer are the same, so, there is a symmericity. The second set is included to test the methods for non-symmetric cases. There are two swithcover cases: long and short. By short switchovers it is aimed that the server may face with no customers with a nonzero probability. Hence, this makes possible to test the algorithms in this case as well. Long swithcover case eliminates that probability, that is server always faces with some customers. Also, in a test problem both of the switchovers are short or long. But, problems which have non-symmetric switchovers should also be tested. The whole set of problems can be seen in Tables B.1 and B.2 in the Appendix. Now performance of the solution methods will be compared.

4.2.1 Exact and Decomposed

According to Takagi [34], decomposed models represent the system exactly as long as the vacation periods distributions are estimated accurately. Table 4.2 gives the summary results of the absolute percentage error in the performance measures. Absolute errors are calculated as:

$$\left|\frac{EXACT - DECOMPOSED}{EXACT}\right| * 100.$$

First three rows are the average, maximum and minimum absolute errors for the whole 144 problems. THR1 and THR2 represent average thruput rates and AB1 and AB2 represent the average buffer levels of the customer types 1 and 2 respectively. Mean absolute errors are in very acceptable level. Succeeding lines in Table 4.2 give the average absolute errors for fixed buffer size. The results show that the decomposition can be further modified, because errors

Exact vs. Decomposition Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	0.17	0.20	1.56	0.54
Max	0.91	1.00	9.26	4.87
Min	0.00	0.00	0.00	0.00
Buffer Sizes (Avg)				
S=1	0.31	0.24	0.72	0.24
S=2	0.29	0.25	1.08	0.30
S=5	0.18	0.23	1.49	0.46
S=10	0.11	0.19	1.80	0.65
S=15	0.08	0.16	2.04	0.76
S=20	0.05	0.13	2.23	0.84
Switchovers (Avg)				
Short	0.21	0.17	2.27	0.77
Long	0.13	0.23	0.85	0.31
Loads (Avg)				
High and High	0.09	0.06	0.07	0.03
Medium and High	0.14	0.11	0.25	0.05
Light and High	0.13	0.17	1.44	0.17
Medium vs. Medium	0.25	0.29	0.95	0.41
Light and Medium	0.19	0.35	3.55	0.76
Light and Light	0.22	0.22	3.10	1.84

Table 4.2: Decomposition accuracy of the exhaustive case: A summary.

in the thruput rates decreases but errors in the average buffer levels increases as the buffer sizes increase. Also it can easily concluded that errors are more likely to decline as the vacation periods lengths increase. This is because, in the long switchover cases the average of the errors are much less than the one in the short switchover cases. This result is also observed in the case of loads. The instances which cause maximum errors are given as follows:

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S_1, S_2)$	THR1	THR2	AB1	AB2
65	(0.2, 1.0, 0.5, 1.0, 0.1, 0.1, 15, 15)	0.49	**1.00**	1.73	0.54
69	(0.2, 1.0, 0.2, 1.0, 0.1, 0.1, 5, 5)	**0.91**	0.91	0.92	0.92
72	(0.2, 1.0, 0.2, 1.0, 0.1, 0.1, 20, 20)	0.13	0.13	4.87	**4.87**
102	(0.1, 1.0, 0.8, 2.0, 1.0, 2.0, 20, 20)	0.00	0.00	**9.26**	1.10

These problems have a common characteristic of having light or medium load conditions. It is consistent with one of the previous observations: accuracy reduces as the load becomes lighter.

Also, thruputs rates are all overestimated whereas average buffer levels are underestimated by the decomposition. So one can conclude that the vacation periods are underestimated. This problem may be overcome by initializing the facing probabilities with an increasing order in the fixed point algorithm. The detailed results for each problem can be found in Tables B.3- B.5,in Appendix.

	Decomposition				Exact
Buffer Sizes (Avg)	Step	1st Part	2nd Part	Total	
S=1	3.46	0.00	0.00	0.00	0.00
S=2	4.13	0.00	0.01	0.01	0.01
S=5	6.13	0.00	0.03	0.03	0.08
S=10	8.38	0.03	0.16	0.19	0.76
S=15	9.75	0.06	0.51	0.57	2.96
S=20	10.75	0.13	1.29	1.42	8.56

Table 4.3: Computation time requirements of the exhaustive case in cpu. seconds.

Table 4.3 gives the computational comparison between exact and decomposed methods. STEP represents the average number step performed in the fixed point algorithm to get the facing probabilities. 1st PART and 2nd PART represent the average time spent by the fixed point algorithm and sparse solver respectively. TOTAL is simply the sum of the two. Last column, EXACT, is the time spent by the sparse solver for the exact method. Time values are measured in terms of cpu seconds, so the last digits may change. As it is seen

	Exact			Decomposed		
Buffer Size	Size	Nonzero per row	Density (%)	Size	Nonzero per row	Density (%)
S=1	16x16	3.81	23.81	10x10	3.50	35.00
S=2	36x36	4.25	11.81	18x18	4.11	22.83
S=5	144x144	4.65	3.23	54x54	4.87	9.02
S=10	484x484	4.81	0.99	154x154	5.31	3.45
S=15	1024x1024	4.87	0.48	304x304	5.50	1.81
S=20	1764x1764	4.90	0.28	504x504	5.61	1.11

Table 4.4: Problem dimensions of the exact and decomposed models in the exhaustive case.

the difference between the time required for exact and decomposed systems increases as the problem size increases. Also the number of steps performed by the fixed point problem increases as the buffer sizes increase. Table 4.4 shows the dimensions of the infinitesimal generators in exact and decomposed methods. Although in the case of exact method, the generator is more sparse, the dimension of the problem is about four times the one in the decomposition. But, recall that there are two infinitesimal generators to be solved in decomposed cases.

4.2.2 Exact and Approximation

As it is recalled, in approximation method, the steady state thruput rates and the average buffer lengths are found without solving the Markov Chain state probabilities. The accuracy of the approximation is tested using the same set of problems. Table 4.5 gives the summary results of the absolute percentage errors in the performance measures. Mean absolute errors are in very acceptable level. But it seems there is no relation between the error percentage and design parameters except loads, the approximation accuracy decreases as the loads become lighter.

The instances which cause maximum errors are given as follows:

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S_1, S_2)$	THR1	THR2	AB1	AB2
65	(0.2, 1.0, 0.5, 1.0, 0.1, 0.1, 15, 15)	0.49	**1.00**	1.64	0.33
69	(0.2, 1.0, 0.2, 1.0, 0.1, 0.1, 5, 5)	**0.91**	0.91	0.05	0.05
72	(0.2, 1.0, 0.2, 1.0, 0.1, 0.1, 20, 20)	0.13	0.13	4.86	**4.86**
102	(0.1, 1.0, 0.8, 2.0, 1.0, 2.0, 20, 20)	0.00	0.00	**9.26**	1.10

As one notice these are the same problems faced in decomposition with the maximum errors. Hence, the same argument applies here.

Also, thruputs rates are all overestimated whereas average buffer levels are underestimated by the approximation. This result is expected, because of the problem related with the estimation of vacation period's distribution. The detailed results are in Tables B.6- B.8 in Appendix.

Exact vs. Approximation Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	0.17	0.20	1.99	1.08
Max	0.91	1.00	9.26	4.86
Min	0.00	0.00	0.01	0.04
Buffer Sizes (Avg)				
S=1	0.31	0.24	2.30	2.12
S=2	0.29	0.25	1.80	1.24
S=5	0.18	0.23	1.63	0.67
S=10	0.11	0.19	1.86	0.75
S=15	0.08	0.16	2.09	0.83
S=20	0.05	0.13	2.28	0.85
Switchovers (Avg)				
Short	0.21	0.17	2.28	1.24
Long	0.13	0.23	1.71	0.96
Loads (Avg)				
High and High	0.09	0.06	0.73	0.74
Medium and High	0.14	0.11	0.87	0.73
Light and High	0.13	0.17	1.80	0.69
Medium and Medium	0.25	0.29	1.37	0.97
Light and Medium	0.19	0.35	3.86	1.20
Light and Light	0.22	0.22	3.33	2.16

Table 4.5: Approximation accuracy of the exhaustive case: A summary.

Exact vs. Decomposition Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	0.22	0.41	1.40	0.70
Max	1.04	2.79	10.95	8.00
Min	0.00	0.00	0.00	0.00
Buffer Sizes (Avg)				
S=1	0.28	0.32	0.58	0.26
S=2	0.28	0.36	0.94	0.37
S=5	0.19	0.47	1.64	0.74
S=10	0.12	0.49	2.43	1.43
Switchovers (Avg)				
Short	0.36	0.68	2.58	1.30
Long	0.08	0.14	0.21	0.10
Loads (Avg)				
High and High	0.13	0.20	0.07	0.08
Medium and High	0.17	0.24	0.31	0.15
Light and High	0.09	0.43	0.91	0.14
Medium and Medium	0.45	0.65	1.22	0.61
Light and Medium	0.17	0.61	3.00	1.21
Light and Light	0.31	0.33	2.87	2.02

Table 4.6: Decomposition accuracy of the gated case: A summary.

4.3 Gated Case

As it is recalled gated case is more complicated than the exhaustive counterpart. So the problem size is significantly larger when the same parameters are used. In this case the same problem set is used, but the maximum buffer size is set to 10. Therefore, 96 problems are tested. The full set of problems can be seen in Table C.1 in Appendix.

4.3.1 Exact and Decomposition

Remember that the existence of invariant facing probabilities and the convergence of the fixed point algorithm are not proven for the gated case. But, all the test problems converged. Table 4.6 gives the summary results of the absolute errors between the exact and the decomposed models. The average errors are a little bit higher than the exhaustive case. But they are still in acceptable level. The effect of switchover is clear in this case too. Also, when the loads are lighter, that is the vacation periods are shorter the mean absolute errors in average buffer levels increase. But, the errors in estimation of thruputs seem that they are independent of load conditions.

The instances which cause maximum errors are given as follows:

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S_1, S_2)$	THR1	THR2	AB1	AB2
20	(0.2, 1.0, 0.2, 1.0, 1.0, 1.0, 10, 10)	0.02	1.18	**10.95**	7.78
24	(0.2, 1.0, 0.5, 1.0, 1.0, 1.0, 10, 10)	0.00	0.00	8.00	**8.00**
62	(0.4, 1.0, 0.8, 2.0, 1.0, 2.0, 2, 2)	**1.04**	0.93	1.79	0.52
64	(0.4, 1.0, 0.8, 2.0, 1.0, 2.0, 10, 10)	0.40	bf 2.79	9.67	4.77

These problems have a common characteristic of having light or medium load conditions. It is consistent with one of the previous observations: Accuracy in estimation of average buffer level reduces as the load becomes lighter. Notice that the maximum error in thruput is caused by a problems having medium load conditions.

Although decomposed model overestimates the thruput rates in most of the test problems, there is no such observation in the average buffer levels. Decomposed model did not show any trend in the estimation of buffer levels.

The detailed results can be seen in Tables C.2, and C.3 in Appendix.

In addition, the huge savings in the problem size and the computational efforts are realized. As Table 4.7 shows, when the buffer sizes are 10, the exact method finds the solution in 25.8 cpu seconds whereas the decomposed model

	Decomposition				Exact
Buffer Sizes (Avg)	Step	1st Part	2nd Part	Total	
S=1	4.83	0.00	0.00	0.00	0.00
S=2	6.08	0.00	0.01	0.01	0.02
S=5	7.88	0.00	0.03	0.0.3	0.69
S=10	10.17	0.01	0.18	0.19	25.84

Table 4.7: Computation time requirements of the gated case in cpu. seconds..

	Exact			Decomposed		
Buffer Size	Size	Nonzero per row	Density (%)	Size	Nonzero per row	Density (%)
S=1	16x16	3.75	23.45	8x8	3.25	40.63
S=2	48x48	4.21	8.77	17x17	3.76	23.50
S=5	312x312	4.59	1.47	62x62	4.23	6.82
S=10	1672x1672	4.76	0.28	197x197	4.43	2.25

Table 4.8: Problem dimensions of the exact and decomposed models in the gated case.

run only 0.19 cpu seconds. Also the construction phase of the exact method is very time consuming because of the huge size of the infinitesimal generator. Table 4.8 gives the size and the density of the infinitesimal generator for each model. Also, the density of the decomposed model's generators is suitable to run a sparse solver efficiently for moderate buffer sizes.

4.3.2 Exact and Approximate

The accuracy of the approximation is tested using the same set of problems. Table 4.9 gives the summary results of the absolute percentage errors in the performance measures. It seems there is no relation between the approximation accuracy and the design parameters except switchovers. The detailed results of the approximation is given in Tables C.4, and C.5 in Appendix.

Following table gives the instances causing maximum errors. Although two of them are different from the ones found in decomposed model, the conclusion is the same. Thus, approximation behaves similarly as decomposition, in estimating the performance measures.

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S_1, S_2)$	THR1	THR2	AB1	AB2
12	(0.2, 1.0, 0.8, 1.0, 1.0, 1.0, 10, 10)	0.02	2.14	0.09	**9.69**
62	(0.4, 1.0, 0.8, 2.0, 1.0, 2.0, 2, 2)	**1.85**	1.47	1.66	1.68
64	(0.4, 1.0, 0.8, 2.0, 1.0, 2.0, 10, 10)	0.50	**3.21**	9.25	3.47
68	(0.1, 1.0, 0.8, 2.0, 1.0, 2.0, 10, 10)	0.00	0.37	**10.96**	3.66

Exact vs. Approximation Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	0.38	0.62	2.26	2.31
Max	1.85	3.21	10.96	9.69
Min	0.00	0.00	0.00	0.00
Buffer Sizes (Avg)				
S=1	0.28	0.32	0.58	0.27
S=2	0.54	0.65	1.73	1.90
S=5	0.43	0.81	3.01	3.22
S=10	0.27	0.70	3.73	3.84
Switchovers (Avg)				
Short	0.60	1.06	3.63	3.57
Long	0.16	0.19	0.89	1.05
Loads (Avg)				
High and High	0.35	0.23	3.29	2.51
Medium and High	0.44	0.42	1.58	2.82
Light and High	0.15	0.87	0.88	3.39
Medium and Medium	0.74	0.91	2.17	1.64
Light and Medium	0.23	0.91	2.93	1.60
Light and Light	0.37	0.39	2.74	1.88

Table 4.9: Approximation accuracy of the gated case: A summary.

4.4 G-limited Case

In this case one more design parameter is added to the system: Batch size. All the other factors are kept the same. The parameters of the problems are shown in Table 4.10.

1.Buffer Sizes	(S,K)
	(2,1), (5,1), (5,2), (10,1), (10,2), (10,5), (15,1), (15,2), (15,5)
2.Switchovers	β_{ij}
Short	μ_i
Long	$\mu_i/10$
3.Loads	$(\lambda_1, \mu_1, \lambda_2, \mu_2)$
High and High	(0.8,1.0,0.8,1.0), (0.7,1.0,1.4,2.0)
Medium and High	(0.5,1.0,0.8,1.0), (0.4,1.0,1.4,2.0)
Light and High	(0.2,1.0,0.8,1.0), (0.1,1.0,1.4,2.0)
Medium and Medium	(0.5,1.0,0.5,1.0), (0.4,1.0,0.8,2.0)
Light and Medium	(0.2,1.0,0.5,1.0), (0.1,1.0,0.8,2.0)
Light and Light	(0.2,1.0,0.2,1.0), (0.1,1.0,0.2,2.0)

Table 4.10: Problem generation parameters for the G-limited case ($S_1 = S_2 = S, K_1 = K_2 = K$)

The size of the largest test problem is (15,5). The complete set of problems can be seen in Tables D.1, and D.2 in Appendix. The total number of problems tested is 216.

4.4.1 Exact and Decomposition

In the G-limited case the existence of invariant facing probabilities and convergence of the fixed point algorithm are not proven. Although majority of the test problems converge, there are some instances that do not converge after 99 iterations. But these cases are also included in the summary figures, shown in Table 4.11. But the average absolute errors are still competent with the other policies. However, the maximum absolute errors are much higher than

the others. The effect of the switchover is also clear in this policy. But the accuracy of the decomposition seems not related with the buffer and batch sizes. The average error increases when one of the loads is light and the other is light or medium.

The problems causing maximum errors can be given as follows:

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S, K)$	THR1	THR2	AB1	AB2
45	(0.2, 1.0, 0.5, 1.0, 1.0. 1.0, 15, 5)	0.00	1.46	16.1	**28.2**
52	(0.2, 1.0, 0.2, 1.0, 1.0. 1.0, 15, 1)	0.04	0.04	**21.2**	21.2
141	(0.4, 1.0, 0.8, 2.0, 1.0. 2.0, 10, 5)	0.15	**5.62**	5.45	9.80
165	(0.7, 1.0,1.4, 2.0, 0.1. 0.2, 5, 2)	**2.78**	5.56	0.15	0.13

Although maximum errors mostly seen in light load conditions, it is interesting to note an instance which gives maximum error in the estimation of thruput rate of the first customer, and this contradicts the above argument. The instance is high loaded and swithcovers are long. Only one of the non-convergent instances gives the maximum error. Although the fixed point algorithm did not converge for these cases, the facing probabilities (q_{jk}'s) are estimated as accurate as the others. It may be a sign of convergence of the algorithm. Because accuracy of the non-convergent instances is not worse than the convergent one.

Decomposition overestimates thruput rates for all cases, but there is no such observation in the average buffer levels. The detailed results are reported in Tables D.3- D.6 in Appendix.

In spite of the reduction in accuracy, significant savings in problem dimension and computational efforts are realized. As Table 4.12 shows the largest problems are solved, on the average, in 99.8 cpu seconds by the exact method whereas it is only 0.25 cpu seconds by the decomposition method. It is interesting to note that the average number of steps performed by the fixed point algorithm decreases as the batch size increases when the buffer size is kept constant. Table 4.13 gives the size and the density of the infinitesimal generator for both models. The saving can also be seen in these figures. The

Exact vs. Decomposition Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	0.04	0.45	1.22	1.31
Max	2.78	5.62	21.25	28.20
Min	0.00	0.00	0.00	0.00
Buffer Sizes (Avg)				
S=2, K=1	0.11	0.43	0.51	0.45
S=5, K=1	0.03	0.24	0.62	0.68
S=5, K=2	0.15	0.80	1.11	1.02
S=10, K=1	0.01	0.19	0.95	1.05
S=10, K=2	0.00	0.48	1.03	1.01
S=10, K=5	0.02	0.69	2.37	2.33
S=15, K=1	0.00	0.19	1.10	1.19
S=15, K=2	0.00	0.50	1.00	0.98
S=15, K=5	0.01	0.56	2.30	3.07
Switchovers (Avg)				
Short	0.07	0.58	3.09	3.26
Long	0.03	0.09	0.04	0.09
Loads (Avg)				
High and High	0.08	0.19	0.00	0.01
Medium and High	0.00	0.47	0.01	0.03
Light and High	0.00	0.61	0.05	0.06
Medium and Medium	0.03	0.64	0.31	0.65
Light and Medium	0.02	0.64	1.89	2.00
Light and Light	0.08	0.14	5.07	5.10

Table 4.11: Decomposition accuracy of the G-limited case: A summary.

	Decomposition				Exact
Buffer Sizes (Avg)	Step	1st Part	2nd Part	Total	
S=2, K=1	9.25	0.00	0.00	0.00	0.01
S=5, K=1	18.29	0.01	0.01	0.02	0.10
S=5, K=2	13.21	0.01	0.01	0.02	0.25
S=10, K=1	30.08	0.03	0.01	0.04	1.38
S=10, K=2	23.83	0.03	0.04	0.07	4.24
S=10, K=5	16.08	0.03	0.09	0.12	10.63
S=15, K=1	35.08	0.06	0.03	0.09	6.60
S=15, K=2	32.13	0.07	0.07	0.13	16.89
S=15, K=5	22.25	0.07	0.18	0.25	99.86

Table 4.12: Computational results of the G-limited case.

density of the decomposed generators are also suitable to run a sparse solver efficiently for moderate buffer sizes, although they are more dense than their exact counterparts.

4.4.2 Exact and Approximate

The accuracy of the approximation is tested using the same set of problems. Table 4.14 gives the summary results of the absolute percentage errors in the performance measures.

The average errors are a little bit higher, especially in average buffer levels. But the instances that do not converge make them worse. The problem size and accuracy seem not related. Nonetheless, switchover and loads effects on the performance of the approximation are similar as in the gated case.

	Exact			Decomposed		
Buffer Size	Size	Nonzero per row	Density (%)	Size	Nonzero per row	Density (%)
S=2, K=1	36x36	4.22	11.72	12x12	3.58	29.83
S=5, K=1	144x144	4.64	3.22	24x24	3.92	16.33
S=5, K=2	204x204	4.64	2.28	35x35	4.06	11.60
S=10, K=1	484x484	4.81	0.99	44x44	4.07	9.25
S=10, K=2	704x704	4.81	0.68	65x65	4.18	6.43
S=10, K=5	1232x1232	4.80	0.39	122x122	4.32	3.54
S=15, K=1	1024x1024	4.87	0.48	64x64	4.12	6.44
S=15, K=2	1504x1504	4.87	0.32	95x95	4.23	4.45
S=15, K=5	2752x2752	4.87	0.18	182x182	4.35	2.39

Table 4.13: Problem dimensions of the exact and decomposed models in the G-limited case.

The problems causing maximum errors can be given as follows:

NO	$(\lambda_1, \mu_1, \lambda_2, \mu_2, \beta_{12}, \beta_{21}, S, K)$	THR1	THR2	AB1	AB2
51	(0.2, 1.0, 0.2, 1.0, 1.0, 1.0, 10, 5)	**12.26**	**12.26**	20.1	20.1
52	(0.2, 1.0, 0.2, 1.0, 1.0, 1.0, 15, 1)	7.99	7.99	**70.2**	**70.2**

It is expected that cases which give maximum errors have light loads and short switchovers. But instance 51 is not a non-convergent case. Therefore similar arguments in decomposed model apply here. But unlike the decomposed model, approximation does not show any particular over/underestimation in performance measures. The detailed figures can be found in Tables D.7- D.10 in Appendix.

Exact vs. Approximation Absolute Error (%)				
All Cases	THR1	THR2	AB1	AB2
Avg	2.30	1.19	6.79	3.62
Max	12.26	12.26	70.25	70.25
Min	0.00	0.00	0.00	0.00
Buffer Sizes (Avg)				
S=2, K=1	1.87	0.71	3.80	1.46
S=5, K=1	1.77	0.74	5.93	2.57
S=5, K=2	2.56	1.31	5.88	2.80
S=10, K=1	1.83	0.78	7.81	3.68
S=10, K=2	2.58	1.28	8.08	2.60
S=10, K=5	2.86	1.95	5.58	5.78
S=15, K=1	1.82	0.78	8.77	3.95
S=15, K=2	2.62	1.28	9.62	2.40
S=15, K=5	2.76	1.91	5.64	7.35
Switchovers (Avg)				
Short	3.76	2.30	13.19	8.90
Long	0.20	0.03	0.81	0.31
Loads (Avg)				
High and High	0.03	0.03	0.69	0.47
Medium and High	0.80	0.47	4.58	0.52
Light and Medium	3.54	0.65	7.50	0.73
Medium and Medium	0.88	0.69	5.00	1.12
Light and Medium	3.82	1.17	9.34	5.55
Light and Light	4.70	4.15	13.64	13.33

Table 4.14: Approximation accuracy of the G-limited case: A summary.

Chapter 5

Conclusion and Future Research

In this research two-customer finite queue polling model is analyzed for exhaustive, gated, and G-limited service policies. The primary concern is to find the steady state performance measures as the thruput rates and the queue length distributions. In approximation method long run average queue lengths and thruput rates are considered.

Three solution methods are proposed for this purpose. The first one is an exact analysis which requires solution of invariant distribution of the Markov process. Usually this method requires large amount of computational time in the construction of the infinitesimal generator. Although the size of the generator is very large, most of the entries are zero. Therefore, it is possible to use a sparse solver to emancipate the large computational and memory requirements.

The second solution method is a decomposition of the system into two one-server queues with vacations. Vacation periods are scheduled to these servers to imitate the original system. It is shown that vacation periods have a distribution of phase type for all service policies. The primary concern, here, is to find the initial probability vector of these vacation periods. The existence of a

stationary probability vector is important, because it maintains the existence of stationary vacation distributions. According to Takagi [34] the decomposed models exactly represent the real system as long as there exist stationary vacation time distributions. The existence of the stationary probability vector is proven for only the exhaustive case. To find these probabilities a fixed point algorithm is proposed. The convergence of this algorithm is proved again only for exhaustive case. But the algorithm converges for all problems tested in gated case, and fail to converge in 7 of 216 problems in G-limited case. Since the vacation periods are of phase type a Markov Process can be found to model the decomposed systems.

Third method is an approximation that uses the regenaritive property of the system. By this method the steady state thruput rates and the average buffer levels are computed accurately. Note that this method does not give the queue length distribution. The numerical experiences concerning the accuracy of the performance measures and the computational requirements are reported. It is seen that the computational savings gained by the decomposition and approximation are significant.

5.1 Other Performance Measures

Notice that the accuracy of the methods are compared through only the steady state thruput rates and average buffer levels. But there are other performance measures that would be of interest. Among them are waiting time distribution in the queues, mean waiting time, service level and proportion of time the server exercises switchovers. But one can also find these performance measures by using the thruput rates and the queue length distribution.

Let N_j and W_j be the steady state distribution of queue length and waiting time in the queue of the jth type customer respectively. Blanc [3] proves that,

$$E[e^{-\lambda_j(1-z)W_j}] = (1 + (1-z)\frac{\lambda_j}{\mu_j})E[z^{N_j}], \text{ for } |z| \leq 1$$

Therefore, one can utilize the L-S transform of the waiting time distribution. Mean waiting time can easily be deduced from the above equation. Service level can be defined as the proportion of customers that joins the queue, that is complement of the blocking probability. According to Takagi [34],

$$PB_j = 1 - \frac{TH_j}{\lambda_j},$$

where PB_j is the blocking probability of the jth type customer. Steady state proportion of time the server exercises switchovers is a very simple one as,

$$SW = 1 - \sum_{j=1,2} \frac{TH_j}{\mu_j}.$$

5.2 M/G/1 Polling Models

As explained previously, Takagi [34]'s work gives an exact analysis of polling models with generally distributed service and switchover distributions, under three service policies: Exhaustive, gated, and G-limited. Tran-Gia and Raith [39] study on an approximation of 1-limited case. They approximate the vacation periods with two stage phase-type distributions by matching first two moments. The same type of approximation can be done for the other three policies.

It is previously shown that

$$\gamma_j = \begin{cases} \gamma_j^{(0)} & \text{with probability } q_{j0} \\ \gamma_j^{(1)} & \text{with probability } q_{j1} \\ \vdots & \\ \gamma_j^{(k)} & \text{with probability } q_{jk} \\ \vdots & \\ \gamma_j^{(S_j)} & \text{with probability } q_{jS_j} \end{cases} \tag{5.1}$$

If the distributions of $\gamma_j^{(k)}$ are found, the probabilities q_{jk}'s can be found

by a fixed point algorithm also. Let Y_j be a random variable representing the processing time of jth customer, with distribution G_j. For the gated case $\gamma_j^{(k)}$ is simply the sum of k number of Y_j's. For G-limited policy it is almost the same as shown below:

$$\gamma_j^{(k)} = \begin{cases} \sum_{i=1}^{k} Y_j & k \leq K_j \\ \sum_{i=1}^{K_j} Y_j & k \geq K_j \end{cases}$$

But it is somewhat difficult for the exhaustive case. But following system of equations help to identify the Laplace-Stieltjes (L-S) transform of the distribution of $\gamma_j^{(k)}$:

$$\begin{array}{lcllllll}
\gamma_j^{(1)} & = & Y_j + & r_{j1}\gamma_j^{(1)} + & r_{j2}\gamma_j^{(2)} + & \ldots + & r_{j,S_j-1}\gamma_j^{(S_j-1)} + & \tilde{r}_{j,S_j}\gamma_j^{(S_j)} \\
\gamma_j^{(2)} & = & Y_j + & r_{j0}\gamma_j^{(1)} + & r_{j1}\gamma_j^{(2)} + & \ldots + & r_{j,S_j-2}\gamma_j^{(S_j-1)} + & \tilde{r}_{j,S_j-1}\gamma_j^{(S_j)} \\
\gamma_j^{(3)} & = & Y_j + & & r_{j0}\gamma_j^{(2)} + & \ldots + & r_{j,S_j-3}\gamma_j^{(S_j-1)} + & \tilde{r}_{j,S_j-2}\gamma_j^{(S_j)} \\
\vdots & & & & & & & \\
\gamma_j^{(S_j)} & = & Y_j + & & & & r_{j0}\gamma_j^{(S_j-1)} + & \tilde{r}_{j1}\gamma_j^{(S_j)},
\end{array}$$

where $r_{jk} = \int_{t=0}^{\infty} e^{-\lambda_j t}\frac{(\lambda_j t)^k}{k!} dG_j(t)$ and $\tilde{r}_{jk} = \sum_{i=k}^{\infty} r_{ji}$. That is r_{jk}'s are the distribution of Poisson arrivals in a processing time of a customer. After finding the L-S transform of the $\gamma_j^{(k)}$s, it is a simple matter to find the L-S transform of the γ_j, and the first two moments.

5.3 Other Future Directions

An immediate future research would be to prove the convergence of the decomposed systems for the gated and G-limited service policies, or finding new convergent decomposed models. Increasing the number of customers extends this work. Although Takagi [35] gives a unified approach of M/G/1 polling models, he does not provide any numerical analysis. It is still open to be performed. This kind of study can help researchers to compare their approximations with the exact figures.

Some more general service and polling policies can also be analyzed. But, there is a lack in optimization models especially in the presence of finite buffers and nonzero switchovers. Objectives may include maximizing weighted average of thruput rates and minimizing total discounted costs.

Bibliography

[1] Albores, F.X. and P.P. Bocharov, "Two Finite Queues with Relative Priority in a Single Server System with Phase Type Distributions", *Automation and Remote Control*, **54**, 4, 615-623 (1993).

[2] Altıok, T. and G.A. Shiue, "Single-Stage, Multi-Product, Production/Inventory Systems with Backorders", *IIE Transactions*, **26**, 2, 52-61 (1994).

[3] Blanc, J.P.C., "A Numerical Approach to Cyclic Queueing Models", *Queueing Systems*, **6**, 173-188 (1990).

[4] Blondia, C., "A Finite Capacity Multi-Queueing Syatem with Priorities and with Repeated Server Vacations", *Queueing Systems*, **5**, 313-330 (1989).

[5] Boxma, B.J., W.P. Groenendijk, and J.A. Weststrate, "A Pseudoconservation Law for Service Systems with a Polling Table", *IEEE Transactions on Communication*, **38**, 10, 1865-1870 (1990).

[6] Boxma, B.J., H. Levy, and J.A. Weststrate, "Efficient Visit Frequencies for Polling Tables: Minimization of Waiting Cost", *Queueing Systems*, **9**, 133-162 (1991).

[7] Campbell, G.M., "Cyclical Queueing Systems", *European Journal of Operations Research*, **51**, 155-167 (1991).

[8] Cohen, J.W., "A Two Queue, One Server Model with Priority for the Longer Queue", *Queueing Systems*, **2**, 261-283 (1987).

[9] Çınlar, E., *Introduction to Stochastic Processes*, Prentice Hall, N.J. (1975).

[10] Durr, L., "Priority Queues with Random Order of Service", *Operations Researh*, **19**, 453-460 (1971).

[11] Eisenberg, M., "Two Queues with Changeover Times", *Operations Research*, **19**, 386-401 (1971).

[12] Eisenberg, M., "Queues with Periodic Service and Changeover Times", *Operations Research*, **20**, 440-451 (1972).

[13] Fuhrmann, S.W. and R.B. Cooper, "Stochastic Decomposition in the M/G/1 Queue with Generalized Vacations", *Operations Research*, **33**, 1117-1129 (1985).

[14] Fuhrmann, S.W. and A. Moon, "Queues Served in Cyclic Order with and Arbitrary Start-up Distribution", *Naval Research Logistics*, **37**, 123-133 (1990).

[15] Giannakouros, N.P. and A. Laloux, "Waiting-Time Approximation for Service Systems with Star Polling Sequence and Mixed Service Startegies", *IEEE Transactions Communication*, **39**, 7, 1041-1045 (1991).

[16] Hofri, M. and K.W. Ross, "On the Optimal Control of Two Queues with Server Setup Times and its Analysis", *SIAM Journal of Computing*, **16**, 2, 399-420 (1987).

[17] Ibe, O.C. and K.S. Trivedi, "Two Queues with Alternating Service and Server Breakdown", *Queueing Systems*, **7**, 253-268 (1990).

[18] Kundert, K.S. and A.S. Vincetelli, *Sparse User's Guide: A Sparse Linear Equation Solve*, Berkeley (1988).

[19] Larson, H.J., *Introduction to Probability Theory and Statistical Inference*, John Wiley, 3rd Edition, New York (1982).

[20] Leung, K.K., "Cyclic-Service System with Non-preemptive Time-Limited Service", *IEEE Transactions on Communication*, **42**, 8, 2521-2524 (1994).

[21] Leung,Y.T. and R. Suri, "Performance Evaluation of Discrete Manufacturing Systems", *IEEE Control Systems Magazine*, June, 77-86 (1990).

[22] Levy, H., "Binomiall-Gated Service: A Method for Effective Operation and Optimization of Polling Systems", *IEEE Transactions on Communication*, **39**, 9, 1341-1350 (1991).

[23] Manfield, D.R. "Analysis of a Priority Polling System for Two Way Traffic", *IEEE Transactions on Communication*, **33**, 9, 1001-1006 (1985).

[24] Mukherji, B., C.K. Kwok, A.C. Lantz, and W.L. Melody Moh, "Comments on 'Exact Analysis of Asymmetric Polling Systems with Single Buffers' ", *IEEE Transactions on Communication*, **38**, 7, 944-946 (1990).

[25] Neuts, M. F., *Matrix-geometric Solutions in Stochastic Models : An Algorithmic Approach*, Johns Hopkins University Press, Baltimore (1981).

[26] Ross, S.M., *Stochastic Processes*, John Wiley & Sons. Inc (1983).

[27] Rosberg, Z. and P. Kermani, "Customer Scheduling Under Queueing Constraints", *IEEE Transactions on Automatic Control*, **37**, 2, 252-257 (1992).

[28] Sarkar,D. and W.I. Zangwill, "Expected Waiting Time for Nonsymmetric Cyclic Queueing Systems-Exact Results and Applications", *Management Systems*, **35**, 12, 1463-1474 (1989).

[29] Sarkar,D. and W.I. Zangwill, "Variance Effects in Cyclic Production Systems", *Management Systems*, **37**, 4, 444-453 (1991).

[30] Skinner, C.E., "A Priority Queueing System with Server Working Time", *Operations Research*, **14**, 278-285 (1966).

[31] Srinivasan, M.M., "Nondeterministic Polling Systems", *Management Science*, **37**, 6, 667-681 (1991).

[32] Suk, J.B. and C.G. Cassandras, "Optimal Scheduling of Two Competing Queues with Blocking", *IEEE Transactions on Communication*, **36**, 9, 1086-1091 (1991).

[33] Sykes, J.S. "Simplified Analysis of an Alternating Priority Queueing Model with Setup Times". *Operations Research*, **18**, 1182-1192 (1970).

[34] Takagi, H., "Priority Queues with Setup Times". *Operations Research*, **38**, 4, 667-677 (1990).

[35] Takagi, H., "Analysis of Finite-Capacity Polling Systems", *Advances in Applied Probability*, **23**, 373-387 (1991).

[36] Takagi, H. and M. Murata, "Queueing Analysis of Sacan Type TDM and Polling Systems", *Computer Networking and Performance Evaluation*, T.Hasegawa, H.Takagi and Y.Takahashi (editors), Elsevier Science Publishers B.V. (North Holland), 199-211 (1986).

[37] Takine, T. Y. Takahashi, and T. Hasegawa, "Exact Analysis of Asymmetric Polling Systems with Single Buffers", *IEEE Transactions on Communication*, **36**, 10, 1119-1127 (1988).

[38] Tran-Gia, P., "Analysis of Polling Systems with General Input Process and Finite Capacity", *IEEE Transactions on Communication*, **40**, 2, 337-344 (1992).

[39] Tran-Gia, P. and T. Raith, "Multiqueue Systems with Finite Capacity and Nonexhaustive Service", *Computer Networking and Performance Evaluation*, T.Hasegawa, H.Takagi and Y.Takahashi (editors), Elsevier Science Publishers B.V. (North Holland), 213-225 (1986).

[40] Watson, K.S., "Performance Evaluation of Cyclic Service Strategies-A Survey", *Performance'84*, E.Gelenbe (ed.), Elsevier Science Publishers B.V. (North Holland) (1984).

Appendix A

Detailed Numerical Results

All the cases are coded on SUN-Workstation/4.25 with a 64MB ram. The complete lists of MATLAB script files and C codes can be obtained from :

Abdullah Daşcı
Department of Industrial Engineering
Bilkent University
06533 Maltepe Ankara, Turkey
e-mail: dasci@bilkent.edu.tr

Notes

i, j	are the customer indices, $i, j = 1, 2$,
NO	denotes the problem number,
λj	denotes the arrival rate of jth customer,
μj	denotes the processing rate of jth customer,
βij	denotes the rate of the switchover from customer i to customer j,
Sj	denotes the buffer capacity of j th customer,
Kj	denotes the batch size of j th customer,
THRj	denotes the steady state thruput rate of j th customer,
ABj	denotes the average buffer level of j th customer,

Appendix B

Exhaustive Case

NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2	NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2
1	0.8	1	0.8	1	1	1	1	1	37	0.8	1	0.8	1	0.1	0.1	1	1
2	0.8	1	0.8	1	1	1	2	2	38	0.8	1	0.8	1	0.1	0.1	2	2
3	0.8	1	0.8	1	1	1	5	5	39	0.8	1	0.8	1	0.1	0.1	5	5
4	0.8	1	0.8	1	1	1	10	10	40	0.8	1	0.8	1	0.1	0.1	10	10
5	0.8	1	0.8	1	1	1	15	15	41	0.8	1	0.8	1	0.1	0.1	15	15
6	0.8	1	0.8	1	1	1	20	20	42	0.8	1	0.8	1	0.1	0.1	20	20
7	0.5	1	0.8	1	1	1	1	1	43	0.5	1	0.8	1	0.1	0.1	1	1
8	0.5	1	0.8	1	1	1	2	2	44	0.5	1	0.8	1	0.1	0.1	2	2
9	0.5	1	0.8	1	1	1	5	5	45	0.5	1	0.8	1	0.1	0.1	5	5
10	0.5	1	0.8	1	1	1	10	10	46	0.5	1	0.8	1	0.1	0.1	10	10
11	0.5	1	0.8	1	1	1	15	15	47	0.5	1	0.8	1	0.1	0.1	15	15
12	0.5	1	0.8	1	1	1	20	20	48	0.5	1	0.8	1	0.1	0.1	20	20
13	0.2	1	0.8	1	1	1	1	1	49	0.2	1	0.8	1	0.1	0.1	1	1
14	0.2	1	0.8	1	1	1	2	2	50	0.2	1	0.8	1	0.1	0.1	2	2
15	0.2	1	0.8	1	1	1	5	5	51	0.2	1	0.8	1	0.1	0.1	5	5
16	0.2	1	0.8	1	1	1	10	10	52	0.2	1	0.8	1	0.1	0.1	10	10
17	0.2	1	0.8	1	1	1	15	15	53	0.2	1	0.8	1	0.1	0.1	15	15
18	0.2	1	0.8	1	1	1	20	20	54	0.2	1	0.8	1	0.1	0.1	20	20
19	0.5	1	0.5	1	1	1	1	1	55	0.5	1	0.5	1	0.1	0.1	1	1
20	0.5	1	0.5	1	1	1	2	2	56	0.5	1	0.5	1	0.1	0.1	2	2
21	0.5	1	0.5	1	1	1	5	5	57	0.5	1	0.5	1	0.1	0.1	5	5
22	0.5	1	0.5	1	1	1	10	10	58	0.5	1	0.5	1	0.1	0.1	10	10
23	0.5	1	0.5	1	1	1	15	15	59	0.5	1	0.5	1	0.1	0.1	15	15
24	0.5	1	0.5	1	1	1	20	20	60	0.5	1	0.5	1	0.1	0.1	20	20
25	0.2	1	0.5	1	1	1	1	1	61	0.2	1	0.5	1	0.1	0.1	1	1
26	0.2	1	0.5	1	1	1	2	2	62	0.2	1	0.5	1	0.1	0.1	2	2
27	0.2	1	0.5	1	1	1	5	5	63	0.2	1	0.5	1	0.1	0.1	5	5
28	0.2	1	0.5	1	1	1	10	10	64	0.2	1	0.5	1	0.1	0.1	10	10
29	0.2	1	0.5	1	1	1	15	15	65	0.2	1	0.5	1	0.1	0.1	15	15
30	0.2	1	0.5	1	1	1	20	20	66	0.2	1	0.5	1	0.1	0.1	20	20
31	0.2	1	0.2	1	1	1	1	1	67	0.2	1	0.2	1	0.1	0.1	1	1
32	0.2	1	0.2	1	1	1	2	2	68	0.2	1	0.2	1	0.1	0.1	2	2
33	0.2	1	0.2	1	1	1	5	5	69	0.2	1	0.2	1	0.1	0.1	5	5
34	0.2	1	0.2	1	1	1	10	10	70	0.2	1	0.2	1	0.1	0.1	10	10
35	0.2	1	0.2	1	1	1	15	15	71	0.2	1	0.2	1	0.1	0.1	15	15
36	0.2	1	0.2	1	1	1	20	20	72	0.2	1	0.2	1	0.1	0.1	20	20

Table B.1: Exhaustive case: Problem Data

NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2	NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2
73	0.7	1	1.4	2	1	2	1	1	109	0.7	1	1.4	2	0.1	0.2	1	1
74	0.7	1	1.4	2	1	2	2	2	110	0.7	1	1.4	2	0.1	0.2	2	2
75	0.7	1	1.4	2	1	2	5	5	111	0.7	1	1.4	2	0.1	0.2	5	5
76	0.7	1	1.4	2	1	2	10	10	112	0.7	1	1.4	2	0.1	0.2	10	10
77	0.7	1	1.4	2	1	2	15	15	113	0.7	1	1.4	2	0.1	0.2	15	15
78	0.7	1	1.4	2	1	2	20	20	114	0.7	1	1.4	2	0.1	0.2	20	20
79	0.4	1	1.4	2	1	2	1	1	115	0.4	1	1.4	2	0.1	0.2	1	1
80	0.4	1	1.4	2	1	2	2	2	116	0.4	1	1.4	2	0.1	0.2	2	2
81	0.4	1	1.4	2	1	2	5	5	117	0.4	1	1.4	2	0.1	0.2	5	5
82	0.4	1	1.4	2	1	2	10	10	118	0.4	1	1.4	2	0.1	0.2	10	10
83	0.4	1	1.4	2	1	2	15	15	119	0.4	1	1.4	2	0.1	0.2	15	15
84	0.4	1	1.4	2	1	2	20	20	120	0.4	1	1.4	2	0.1	0.2	20	20
85	0.1	1	1.4	2	1	2	1	1	121	0.1	1	1.4	2	0.1	0.2	1	1
86	0.1	1	1.4	2	1	2	2	2	122	0.1	1	1.4	2	0.1	0.2	2	2
87	0.1	1	1.4	2	1	2	5	5	123	0.1	1	1.4	2	0.1	0.2	5	5
88	0.1	1	1.4	2	1	2	10	10	124	0.1	1	1.4	2	0.1	0.2	10	10
89	0.1	1	1.4	2	1	2	15	15	125	0.1	1	1.4	2	0.1	0.2	15	15
90	0.1	1	1.4	2	1	2	20	20	126	0.1	1	1.4	2	0.1	0.2	20	20
91	0.4	1	0.8	2	1	2	1	1	127	0.4	1	0.8	2	0.1	0.2	1	1
92	0.4	1	0.8	2	1	2	2	2	128	0.4	1	0.8	2	0.1	0.2	2	2
93	0.4	1	0.8	2	1	2	5	5	129	0.4	1	0.8	2	0.1	0.2	5	5
94	0.4	1	0.8	2	1	2	10	10	130	0.4	1	0.8	2	0.1	0.2	10	10
95	0.4	1	0.8	2	1	2	15	15	131	0.4	1	0.8	2	0.1	0.2	15	15
96	0.4	1	0.8	2	1	2	20	20	132	0.4	1	0.8	2	0.1	0.2	20	20
97	0.1	1	0.8	2	1	2	1	1	133	0.1	1	0.8	2	0.1	0.2	1	1
98	0.1	1	0.8	2	1	2	2	2	134	0.1	1	0.8	2	0.1	0.2	2	2
99	0.1	1	0.8	2	1	2	5	5	135	0.1	1	0.8	2	0.1	0.2	5	5
100	0.1	1	0.8	2	1	2	10	10	136	0.1	1	0.8	2	0.1	0.2	10	10
101	0.1	1	0.8	2	1	2	15	15	137	0.1	1	0.8	2	0.1	0.2	15	15
102	0.1	1	0.8	2	1	2	20	20	138	0.1	1	0.8	2	0.1	0.2	20	20
103	0.1	1	0.2	2	1	2	1	1	139	0.1	1	0.2	2	0.1	0.2	1	1
104	0.1	1	0.2	2	1	2	2	2	140	0.1	1	0.2	2	0.1	0.2	2	2
105	0.1	1	0.2	2	1	2	5	5	141	0.1	1	0.2	2	0.1	0.2	5	5
106	0.1	1	0.2	2	1	2	10	10	142	0.1	1	0.2	2	0.1	0.2	10	10
107	0.1	1	0.2	2	1	2	15	15	143	0.1	1	0.2	2	0.1	0.2	15	15
108	0.1	1	0.2	2	1	2	20	20	144	0.1	1	0.2	2	0.1	0.2	20	20

Table B.2: Exhaustive case: Problem Data *(continued)*

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.29800	0.29800	0.62749	0.62749	0.29929	0.29929	0.62588	0.62588	0.43	0.43	0.26	0.26
2	0.38939	0.38939	1.29159	1.29159	0.39056	0.39056	1.28988	1.28988	0.30	0.30	0.13	0.13
3	0.46370	0.46370	3.41027	3.41027	0.46396	0.46396	3.41023	3.41023	0.06	0.06	0.00	0.00
4	0.48593	0.48593	7.03487	7.03487	0.48594	0.48594	7.03513	7.03513	0.00	0.00	0.00	0.00
5	0.49165	0.49165	10.6237	10.6237	0.49165	0.49165	10.6238	10.6238	0.00	0.00	0.00	0.00
6	0.49410	0.49410	14.1650	14.1650	0.49410	0.49410	14.1651	14.1651	0.00	0.00	0.00	0.00
7	0.23492	0.31856	0.53015	0.60179	0.23626	0.32020	0.52746	0.59974	0.57	0.52	0.51	0.34
8	0.29805	0.43650	1.08937	1.20026	0.29959	0.43825	1.08494	1.19763	0.52	0.40	0.41	0.22
9	0.33158	0.56752	2.94209	3.00228	0.33218	0.56822	2.93859	3.00140	0.18	0.12	0.12	0.03
10	0.32403	0.63478	6.33852	5.99828	0.32414	0.63493	6.33778	5.99868	0.03	0.02	0.01	0.01
11	0.31507	0.66084	9.88195	8.96052	0.31508	0.66087	9.88219	8.96123	0.01	0.01	0.00	0.01
12	0.30897	0.67424	13.4694	11.8807	0.30897	0.67426	13.4698	11.8813	0.00	0.00	0.00	0.01
13	0.13411	0.34823	0.32941	0.56470	0.13494	0.34992	0.32529	0.56259	0.61	0.49	1.25	0.37
14	0.16162	0.49550	0.65531	1.08024	0.16255	0.49764	0.64569	1.07678	0.57	0.43	1.47	0.32
15	0.17430	0.66147	1.76798	2.51619	0.17471	0.66288	1.75062	2.51079	0.24	0.21	0.98	0.21
16	0.17574	0.73704	3.79370	4.67667	0.17592	0.73760	3.77361	4.66632	0.10	0.08	0.53	0.22
17	0.17722	0.76236	5.76946	6.62141	0.17733	0.76263	5.74757	6.60686	0.07	0.03	0.38	0.22
18	0.17915	0.77401	7.64590	8.42475	0.17924	0.77416	7.62267	8.40751	0.05	0.02	0.30	0.20
19	0.25000	0.25000	0.50000	0.50000	0.25167	0.25167	0.49664	0.49664	0.67	0.67	0.67	0.67
20	0.33325	0.33325	0.96879	0.96879	0.33544	0.33544	0.96232	0.96232	0.65	0.65	0.67	0.67
21	0.41364	0.41364	2.25589	2.25589	0.41506	0.41506	2.24422	2.24422	0.34	0.34	0.52	0.52
22	0.44973	0.44973	4.16249	4.16249	0.45034	0.45034	4.14416	4.14416	0.14	0.14	0.44	0.44
23	0.46391	0.46391	5.94459	5.94459	0.46425	0.46425	5.92264	5.92263	0.07	0.07	0.37	0.37
24	0.47168	0.47168	7.67258	7.67258	0.47189	0.47189	7.64871	7.64870	0.05	0.05	0.31	0.31
25	0.13999	0.27102	0.30004	0.45796	0.14094	0.27265	0.29525	0.45469	0.68	0.60	1.60	0.71
26	0.17306	0.37430	0.52870	0.81789	0.17415	0.37656	0.51672	0.81076	0.63	0.60	2.27	0.87
27	0.19401	0.47199	0.98103	1.50838	0.19441	0.47343	0.95181	1.48769	0.20	0.31	2.98	1.37
28	0.19914	0.49698	1.26047	1.91768	0.19920	0.49726	1.21624	1.87657	0.03	0.06	3.51	2.14
29	0.19987	0.49963	1.31781	2.00172	0.19988	0.49967	1.26898	1.95355	0.01	0.01	3.70	2.41
30	0.19998	0.49995	1.32794	2.01685	0.19998	0.49995	1.27814	1.96723	0.00	0.00	3.75	2.46
31	0.14757	0.14757	0.26211	0.26211	0.14841	0.14841	0.25793	0.25793	0.57	0.57	1.60	1.60
32	0.18376	0.18376	0.39416	0.39416	0.18459	0.18459	0.38436	0.38436	0.45	0.45	2.49	2.49
33	0.19943	0.19943	0.49307	0.49307	0.19953	0.19953	0.47444	0.47444	0.05	0.05	3.78	3.78
34	0.19999	0.19999	0.49994	0.49994	0.19999	0.19999	0.47996	0.47996	0.00	0.00	4.00	4.00
35	0.20000	0.20000	0.49999	0.49999	0.20000	0.20000	0.47999	0.47999	0.00	0.00	4.00	4.00
36	0.20000	0.20000	0.50000	0.50000	0.20000	0.20000	0.47999	0.47999	0.00	0.00	4.00	4.00
37	0.07588	0.07588	0.90513	0.90513	0.07589	0.07589	0.90512	0.90512	0.01	0.01	0.00	0.00
38	0.14826	0.14826	1.73574	1.73574	0.14829	0.14829	1.73570	1.73570	0.02	0.02	0.00	0.00
39	0.29323	0.29323	4.01980	4.01980	0.29327	0.29327	4.01975	4.01975	0.01	0.01	0.00	0.00
40	0.39045	0.39045	7.64538	7.64538	0.39046	0.39046	7.64541	7.64541	0.00	0.00	0.00	0.00
41	0.42806	0.42806	11.2089	11.2089	0.42806	0.42806	11.2089	11.2089	0.00	0.00	0.00	0.00
42	0.44688	0.44688	14.7344	14.7344	0.44688	0.44688	14.7345	14.7345	0.00	0.00	0.00	0.00
43	0.06344	0.07687	0.87311	0.90391	0.06345	0.07688	0.87308	0.90388	0.02	0.02	0.00	0.00
44	0.11640	0.15364	1.65867	1.72611	0.11644	0.15370	1.65858	1.72601	0.03	0.04	0.01	0.01
45	0.20579	0.32889	3.80721	3.89942	0.20586	0.32905	3.80698	3.89916	0.04	0.05	0.01	0.01
46	0.25174	0.47857	7.29938	7.10806	0.25179	0.47869	7.29955	7.10827	0.02	0.02	0.00	0.00
47	0.26527	0.54934	10.8331	10.1237	0.26529	0.54941	10.8337	10.1244	0.01	0.01	0.01	0.01
48	0.27109	0.58854	14.3911	13.0507	0.27109	0.58858	14.3918	13.0515	0.00	0.01	0.00	0.01

Table B.3: Exhaustive case: Exact and decomposed

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
49	0.04803	0.07804	0.75981	0.90244	0.04805	0.07809	0.75974	0.90238	0.03	0.07	0.01	0.01
50	0.08202	0.15918	1.37606	1.71611	0.08207	0.15939	1.37571	1.71579	0.06	0.13	0.03	0.02
51	0.13137	0.35739	2.91189	3.80023	0.13153	0.35817	2.90976	3.79877	0.12	0.22	0.07	0.04
52	0.15341	0.53666	5.35786	6.72497	0.15363	0.53777	5.35303	6.72493	0.14	0.21	0.09	0.00
53	0.16009	0.62207	7.84486	9.38310	0.16029	0.62313	7.83977	9.38877	0.13	0.17	0.06	0.06
54	0.16379	0.66882	10.3239	11.9104	0.16397	0.66978	10.3209	11.9239	0.11	0.14	0.03	0.11
55	0.06423	0.06423	0.87153	0.87153	0.06425	0.06425	0.87148	0.87148	0.04	0.04	0.01	0.01
56	0.12057	0.12057	1.64626	1.64626	0.12067	0.12067	1.64601	1.64601	0.08	0.08	0.02	0.02
57	0.23300	0.23300	3.64322	3.64322	0.23337	0.23337	3.64197	3.64197	0.16	0.16	0.03	0.03
58	0.32206	0.32206	6.49503	6.49503	0.32267	0.32267	6.49399	6.49399	0.19	0.19	0.02	0.02
59	0.36608	0.36608	9.11089	9.11089	0.36677	0.36677	9.11384	9.11384	0.19	0.19	0.03	0.03
60	0.39241	0.39241	11.6181	11.6181	0.39308	0.39308	11.6279	11.6279	0.17	0.17	0.08	0.08
61	0.04851	0.06516	0.75741	0.86967	0.04854	0.06523	0.75727	0.86954	0.06	0.10	0.02	0.02
62	0.08429	0.12476	1.35754	1.63363	0.08441	0.12505	1.35671	1.63286	0.14	0.24	0.06	0.05
63	0.14384	0.25415	2.66394	3.50578	0.14436	0.25566	2.65612	3.49981	0.36	0.60	0.29	0.17
64	0.17996	0.36820	4.10790	5.83145	0.18092	0.37158	4.07024	5.81185	0.54	0.92	0.92	0.34
65	0.19215	0.42615	5.11364	7.56067	0.19310	0.43042	5.02537	7.52002	0.49	1.00	1.73	0.54
66	0.19681	0.45838	5.84100	8.84950	0.19752	0.46264	5.68725	8.77020	0.36	0.93	2.63	0.90
67	0.04904	0.04904	0.75476	0.75476	0.04912	0.04912	0.75436	0.75436	0.16	0.16	0.05	0.05
68	0.08634	0.08634	1.34037	1.34037	0.08667	0.08667	1.33797	1.33797	0.38	0.38	0.18	0.18
69	0.15090	0.15090	2.49106	2.49106	0.15228	0.15228	2.46803	2.46803	0.91	0.91	0.92	0.92
70	0.18805	0.18805	3.37178	3.37178	0.18963	0.18963	3.28128	3.28128	0.84	0.84	2.68	2.68
71	0.19726	0.19726	3.67601	3.67601	0.19802	0.19802	3.52445	3.52445	0.39	0.39	4.12	4.12
72	0.19940	0.19940	3.76756	3.76756	0.19966	0.19966	3.58425	3.58425	0.13	0.13	4.87	4.87
73	0.34386	0.42911	0.50876	0.69349	0.34583	0.43021	0.50595	0.69270	0.57	0.26	0.55	0.11
74	0.45303	0.57303	1.02621	1.40056	0.45544	0.57370	1.02097	1.40030	0.53	0.12	0.51	0.02
75	0.55518	0.69135	2.61409	3.59731	0.55627	0.69138	2.60818	3.59888	0.20	0.00	0.23	0.04
76	0.59817	0.71971	5.25890	7.34597	0.59836	0.71978	5.25659	7.34724	0.03	0.01	0.04	0.02
77	0.61441	0.72052	7.87042	11.1220	0.61445	0.72054	7.86986	11.1228	0.01	0.00	0.01	0.01
78	0.62357	0.71695	10.4609	14.9062	0.62358	0.71696	10.4609	14.9067	0.00	0.00	0.00	0.00
79	0.24772	0.48061	0.38069	0.65670	0.24935	0.48204	0.37661	0.65568	0.66	0.30	1.07	0.16
80	0.31441	0.69337	0.73463	1.26883	0.31656	0.69496	0.72428	1.26764	0.69	0.23	1.41	0.09
81	0.36150	0.95207	1.77220	2.99415	0.36261	0.95382	1.75052	2.99283	0.31	0.18	1.22	0.04
82	0.37567	1.08243	3.47902	5.71377	0.37597	1.08376	3.45895	5.71152	0.08	0.12	0.58	0.04
83	0.38168	1.12893	5.15797	8.38634	0.38179	1.12974	5.14317	8.38388	0.03	0.07	0.29	0.03
84	0.38558	1.15129	6.82880	11.0752	0.38563	1.15178	6.81808	11.0732	0.01	0.04	0.16	0.02
85	0.08613	0.55997	0.13868	0.60001	0.08647	0.56101	0.13526	0.59927	0.40	0.19	2.47	0.12
86	0.09675	0.85438	0.22754	1.08582	0.09698	0.85604	0.21821	1.08424	0.24	0.19	4.10	0.15
87	0.09972	1.20704	0.46235	2.24564	0.09975	1.20885	0.43997	2.24131	0.03	0.15	4.84	0.19
88	0.09997	1.35165	0.75065	3.48150	0.09997	1.35267	0.71888	3.46881	0.00	0.08	4.23	0.36
89	0.09999	1.38616	0.89634	4.09073	0.09999	1.38661	0.86053	4.06788	0.00	0.03	3.99	0.56
90	0.09999	1.39584	0.95776	4.35465	0.09999	1.39602	0.91995	4.32436	0.00	0.01	3.95	0.70
91	0.25526	0.36418	0.36185	0.54477	0.25732	0.36565	0.35669	0.54292	0.81	0.41	1.42	0.34
92	0.33008	0.50566	0.65098	1.01370	0.33285	0.50772	0.63734	1.01011	0.84	0.41	2.10	0.35
93	0.38683	0.66361	1.22864	2.10224	0.38813	0.66641	1.19030	2.08787	0.34	0.42	3.12	0.68
94	0.39860	0.74567	1.69340	3.20528	0.39877	0.74780	1.63401	3.16371	0.04	0.28	3.51	1.30
95	0.39980	0.77594	1.92309	3.82361	0.39983	0.77713	1.85506	3.76065	0.01	0.15	3.54	1.65
96	0.39997	0.78895	2.05269	4.17840	0.39997	0.78957	1.97949	4.10049	0.00	0.08	3.57	1.86

Table B.4: Exhaustive case: Exact and decomposed *(continued)*

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
97	0.08719	0.41566	0.12806	0.48041	0.08759	0.41663	0.12403	0.47921	0.46	0.23	3.15	0.25
98	0.09777	0.60091	0.18410	0.80068	0.09802	0.60236	0.17373	0.79780	0.25	0.24	5.63	0.36
99	0.09997	0.76875	0.25035	1.27825	0.09998	0.76971	0.22900	1.26960	0.01	0.13	8.52	0.68
100	0.10000	0.79837	0.27130	1.44351	0.10000	0.79850	0.24636	1.42889	0.00	0.02	9.19	1.01
101	0.10000	0.79989	0.27306	1.45875	0.10000	0.79991	0.24780	1.44284	0.00	0.00	9.25	1.09
102	0.10000	0.79999	0.27320	1.46009	0.10000	0.79999	0.24791	1.44402	0.00	0.00	9.26	1.10
103	0.08869	0.15949	0.11306	0.20253	0.08897	0.15976	0.11023	0.20118	0.32	0.17	2.50	0.67
104	0.09863	0.19142	0.13993	0.26346	0.09875	0.19160	0.13461	0.26091	0.12	0.09	3.80	0.97
105	0.09999	0.19991	0.14657	0.28555	0.09999	0.19992	0.13998	0.28213	0.00	0.00	4.50	1.20
106	0.10000	0.20000	0.14662	0.28592	0.10000	0.20000	0.14001	0.28247	0.00	0.00	4.51	1.21
107	0.10000	0.20000	0.14662	0.28592	0.10000	0.20000	0.14001	0.28247	0.00	0.00	4.51	1.21
108	0.10000	0.20000	0.14662	0.28592	0.10000	0.20000	0.14001	0.28247	0.00	0.00	4.51	1.21
109	0.09527	0.09673	0.86389	0.93090	0.09528	0.09675	0.86387	0.93089	0.01	0.02	0.00	0.00
110	0.18324	0.18700	1.62248	1.80796	0.18327	0.18706	1.62243	1.80790	0.02	0.03	0.00	0.00
111	0.35833	0.36903	3.57792	4.27519	0.35838	0.36918	3.57783	4.27503	0.01	0.04	0.00	0.00
112	0.48211	0.50001	6.46697	8.19993	0.48213	0.50013	6.46714	8.19989	0.01	0.02	0.00	0.00
113	0.53464	0.55610	9.20137	12.0494	0.53465	0.55617	9.20169	12.0495	0.00	0.01	0.00	0.00
114	0.56313	0.58626	11.8693	15.8686	0.56314	0.58630	11.8697	15.8688	0.00	0.01	0.00	0.00
115	0.07736	0.09860	0.80658	0.92957	0.07737	0.09863	0.80655	0.92954	0.02	0.03	0.00	0.00
116	0.13964	0.19676	1.47974	1.79789	0.13968	0.19689	1.47960	1.79778	0.03	0.07	0.01	0.01
117	0.24847	0.43114	3.10334	4.15227	0.24859	0.43162	3.10258	4.15174	0.05	0.11	0.02	0.01
118	0.31781	0.65638	5.33818	7.63023	0.31800	0.65704	5.33621	7.63007	0.06	0.10	0.04	0.00
119	0.34611	0.77863	7.37007	10.8508	0.34631	0.77924	7.36728	10.8524	0.06	0.08	0.04	0.01
120	0.36124	0.85437	9.30369	13.9470	0.36141	0.85488	9.30057	13.9507	0.05	0.06	0.03	0.03
121	0.04584	0.10180	0.54159	0.92728	0.04584	0.10186	0.54152	0.92724	0.02	0.06	0.01	0.00
122	0.07062	0.21168	0.85871	1.78244	0.07064	0.21193	0.85827	1.78222	0.03	0.12	0.05	0.01
123	0.09482	0.51294	1.33170	3.98641	0.09486	0.51407	1.32725	3.98495	0.05	0.22	0.33	0.04
124	0.09957	0.84462	1.80104	6.89403	0.09958	0.84703	1.78144	6.88993	0.02	0.28	1.09	0.06
125	0.09994	1.03241	2.22117	9.28738	0.09995	1.03551	2.18010	9.28110	0.01	0.30	1.85	0.07
126	0.09999	1.14671	2.61045	11.3626	0.09999	1.15010	2.54399	11.3535	0.00	0.30	2.55	0.08
127	0.07789	0.08124	0.80527	0.89844	0.07792	0.08129	0.80519	0.89838	0.04	0.06	0.01	0.01
128	0.14214	0.15182	1.46990	1.71971	0.14225	0.15203	1.46949	1.71936	0.08	0.14	0.03	0.02
129	0.26164	0.30054	2.97891	3.89075	0.26215	0.30160	2.97545	3.88829	0.19	0.35	0.12	0.06
130	0.34392	0.44315	4.71546	6.93496	0.34501	0.44558	4.69837	6.92846	0.32	0.55	0.36	0.09
131	0.37521	0.53316	5.98257	9.50813	0.37649	0.53669	5.93832	9.49807	0.34	0.66	0.74	0.11
132	0.38855	0.59756	6.98074	11.7205	0.38971	0.60211	6.89639	11.7034	0.30	0.76	1.21	0.15
133	0.04599	0.08373	0.54008	0.89532	0.04600	0.08383	0.53991	0.89521	0.04	0.11	0.03	0.01
134	0.07108	0.16270	0.84904	1.69915	0.07113	0.16308	0.84801	1.69853	0.07	0.23	0.12	0.04
135	0.09557	0.35419	1.23942	3.67418	0.09566	0.35591	1.22909	3.66943	0.10	0.49	0.83	0.13
136	0.09980	0.55335	1.46337	5.97628	0.09982	0.55662	1.41943	5.95805	0.02	0.59	3.00	0.30
137	0.09999	0.66225	1.61300	7.52307	0.09999	0.66564	1.52685	7.48553	0.00	0.51	5.34	0.50
138	0.10000	0.72268	1.73259	8.58364	0.10000	0.72547	1.60513	8.52295	0.00	0.39	7.36	0.71
139	0.04613	0.05776	0.53866	0.71116	0.04621	0.05789	0.53783	0.71055	0.18	0.21	0.15	0.09
140	0.07139	0.09971	0.84175	1.22419	0.07162	0.10011	0.83741	1.22118	0.33	0.40	0.52	0.25
141	0.09579	0.16652	1.18818	2.05338	0.09608	0.16723	1.16095	2.03748	0.30	0.43	2.29	0.77
142	0.09983	0.19519	1.28749	2.43488	0.09987	0.19542	1.23132	2.40479	0.04	0.12	4.36	1.24
143	0.09999	0.19934	1.30043	2.49882	0.09999	0.19938	1.23629	2.46508	0.00	0.02	4.93	1.35
144	0.10000	0.19991	1.30252	2.50895	0.10000	0.19991	1.23680	2.47452	0.00	0.00	5.05	1.37

Table B.5: Exhaustive case: Exact and decomposed *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.29800	0.29800	0.62749	0.62749	0.29929	0.29929	0.64470	0.64470	0.43	0.43	2.74	2.74
2	0.38939	0.38939	1.29159	1.29159	0.39056	0.39056	1.30995	1.30995	0.30	0.30	1.42	1.42
3	0.46370	0.46370	3.41027	3.41027	0.46396	0.46396	3.43109	3.43109	0.06	0.06	0.61	0.61
4	0.48593	0.48593	7.03487	7.03487	0.48594	0.48594	7.05658	7.05658	0.00	0.00	0.31	0.31
5	0.49165	0.49165	10.6237	10.6237	0.49165	0.49165	10.6456	10.6456	0.00	0.00	0.21	0.21
6	0.49410	0.49410	14.1650	14.1650	0.49410	0.49410	14.1871	14.1871	0.00	0.00	0.16	0.16
7	0.23492	0.31856	0.53015	0.60179	0.23626	0.32020	0.54535	0.61774	0.57	0.52	2.87	2.65
8	0.29805	0.43650	1.08937	1.20026	0.29959	0.43825	1.10461	1.21526	0.52	0.40	1.40	1.25
9	0.33158	0.56752	2.94209	3.00228	0.33218	0.56822	2.96201	3.01695	0.18	0.12	0.68	0.49
10	0.32403	0.63478	6.33852	5.99828	0.32414	0.63493	6.36527	6.01346	0.03	0.02	0.42	0.25
11	0.31507	0.66084	9.88195	8.96052	0.31508	0.66087	9.91224	8.97598	0.01	0.01	0.31	0.17
12	0.30897	0.67424	13.4694	11.8807	0.30897	0.67426	13.5017	11.8962	0.00	0.00	0.24	0.13
13	0.13411	0.34823	0.32941	0.56470	0.13494	0.34992	0.33487	0.57899	0.61	0.49	1.66	2.53
14	0.16162	0.49550	0.65531	1.08024	0.16255	0.49764	0.65520	1.08986	0.57	0.43	0.02	0.89
15	0.17430	0.66147	1.76798	2.51619	0.17471	0.66288	1.76193	2.51724	0.24	0.21	0.34	0.04
16	0.17574	0.73704	3.79370	4.67667	0.17592	0.73760	3.78687	4.67004	0.10	0.08	0.18	0.14
17	0.17722	0.76236	5.76946	6.62141	0.17733	0.76263	5.76104	6.60967	0.07	0.03	0.15	0.18
18	0.17915	0.77401	7.64590	8.42475	0.17924	0.77416	7.63566	8.40985	0.05	0.02	0.13	0.18
19	0.25000	0.25000	0.50000	0.50000	0.25167	0.25167	0.51301	0.51301	0.67	0.67	2.60	2.60
20	0.33325	0.33325	0.96879	0.96879	0.33544	0.33544	0.97774	0.97774	0.65	0.65	0.92	0.92
21	0.41364	0.41364	2.25589	2.25589	0.41506	0.41506	2.25680	2.25680	0.34	0.34	0.04	0.04
22	0.44973	0.44973	4.16249	4.16249	0.45034	0.45034	4.15417	4.15417	0.14	0.14	0.20	0.20
23	0.46391	0.46391	5.94459	5.94459	0.46424	0.46425	5.93098	5.93098	0.07	0.07	0.23	0.23
24	0.47168	0.47168	7.67258	7.67258	0.47189	0.47190	7.65589	7.65589	0.05	0.05	0.22	0.22
25	0.13999	0.27102	0.30004	0.45796	0.14094	0.27265	0.30324	0.46873	0.68	0.60	1.07	2.35
26	0.17306	0.37430	0.52870	0.81789	0.17415	0.37656	0.52236	0.82048	0.63	0.60	1.20	0.32
27	0.19401	0.47199	0.98103	1.50838	0.19441	0.47343	0.95404	1.49027	0.20	0.31	2.75	1.20
28	0.19914	0.49698	1.26047	1.91768	0.19920	0.49726	1.21664	1.87692	0.03	0.06	3.48	2.13
29	0.19987	0.49963	1.31781	2.00172	0.19988	0.49967	1.26903	1.95360	0.00	0.01	3.70	2.40
30	0.19998	0.49995	1.32794	2.01685	0.19998	0.49995	1.27818	1.96720	0.00	0.00	3.75	2.46
31	0.14757	0.14757	0.26211	0.26211	0.14841	0.14841	0.26403	0.26403	0.57	0.57	0.73	0.73
32	0.18376	0.18376	0.39416	0.39416	0.18459	0.18459	0.38686	0.38686	0.45	0.45	1.85	1.85
33	0.19943	0.19943	0.49307	0.49307	0.19953	0.19953	0.47455	0.47455	0.05	0.05	3.76	3.76
34	0.19999	0.19999	0.49994	0.49994	0.19999	0.19999	0.47996	0.47996	0.00	0.00	4.00	4.00
35	0.20000	0.20000	0.49999	0.49999	0.19999	0.19999	0.47997	0.47997	0.00	0.00	4.00	4.00
36	0.20000	0.20000	0.50000	0.50000	0.19999	0.19999	0.47997	0.47997	0.00	0.00	4.01	4.01
37	0.07588	0.07588	0.90513	0.90513	0.07589	0.07589	0.91652	0.91652	0.01	0.01	1.26	1.26
38	0.14826	0.14826	1.73574	1.73574	0.14829	0.14829	1.75332	1.75332	0.02	0.02	1.01	1.01
39	0.29323	0.29323	4.01980	4.01980	0.29327	0.29327	4.04306	4.04306	0.01	0.01	0.58	0.58
40	0.39045	0.39045	7.64538	7.64538	0.39046	0.39046	7.66899	7.66899	0.00	0.00	0.31	0.31
41	0.42806	0.42806	11.2089	11.2089	0.42806	0.42806	11.2320	11.2320	0.00	0.00	0.21	0.21
42	0.44688	0.44688	14.7344	14.7344	0.44688	0.44688	14.7573	14.7573	0.00	0.00	0.16	0.16
43	0.06344	0.07687	0.87311	0.90391	0.06345	0.07688	0.88958	0.91537	0.02	0.02	1.89	1.27
44	0.11640	0.15364	1.65867	1.72611	0.11644	0.15370	1.68339	1.74400	0.03	0.04	1.49	1.04
45	0.20579	0.32889	3.80721	3.89942	0.20586	0.32905	3.83968	3.92354	0.04	0.05	0.85	0.62
46	0.25174	0.47857	7.29938	7.10806	0.25179	0.47869	7.33410	7.13257	0.02	0.03	0.48	0.34
47	0.26527	0.54934	10.8331	10.1237	0.26529	0.54941	10.8689	10.1473	0.01	0.01	0.33	0.23
48	0.27109	0.58854	14.3911	13.0507	0.27109	0.58858	14.4275	13.0734	0.00	0.01	0.25	0.17

Table B.6: Exhaustive case: Exact and approximation

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
49	0.04803	0.07804	0.75981	0.90244	0.04805	0.07809	0.78542	0.91398	0.03	0.07	3.37	1.28
50	0.08202	0.15918	1.37606	1.71611	0.08207	0.15939	1.40796	1.73408	0.06	0.13	2.32	1.05
51	0.13137	0.35739	2.91189	3.80023	0.13153	0.35817	2.94000	3.82312	0.12	0.22	0.97	0.60
52	0.15341	0.53666	5.35786	6.72497	0.15363	0.53777	5.37844	6.74674	0.14	0.21	0.38	0.32
53	0.16009	0.62207	7.84486	9.38310	0.16029	0.62313	7.86334	9.40684	0.13	0.17	0.24	0.25
54	0.16379	0.66882	10.3239	11.9104	0.16397	0.66978	10.3435	11.9391	0.11	0.14	0.19	0.24
55	0.06423	0.06423	0.87153	0.87153	0.06425	0.06425	0.88807	0.88807	0.04	0.04	1.90	1.90
56	0.12057	0.12057	1.64626	1.64626	0.12067	0.12067	1.67112	1.67112	0.08	0.08	1.51	1.51
57	0.23300	0.23300	3.64322	3.64322	0.23337	0.23337	3.67479	3.67479	0.16	0.16	0.87	0.87
58	0.32206	0.32206	6.49503	6.49503	0.32267	0.32267	6.52566	6.52566	0.19	0.19	0.47	0.47
59	0.36608	0.36608	9.11089	9.11089	0.36677	0.36677	9.14229	9.14229	0.19	0.19	0.34	0.34
60	0.39241	0.39241	11.6181	11.6181	0.39308	0.39308	11.6534	11.6534	0.17	0.17	0.30	0.30
61	0.04851	0.06516	0.75741	0.86967	0.04854	0.06523	0.78293	0.88624	0.06	0.10	3.37	1.90
62	0.08429	0.12476	1.35754	1.63363	0.08441	0.12505	1.38853	1.65816	0.14	0.24	2.28	1.50
63	0.14384	0.25415	2.66394	3.50578	0.14436	0.25566	2.68138	3.53131	0.36	0.60	0.65	0.73
64	0.17996	0.36820	4.10790	5.83145	0.18092	0.37158	4.08151	5.83639	0.54	0.92	0.64	0.08
65	0.19215	0.42615	5.11364	7.56067	0.19310	0.43042	5.02996	7.53593	0.49	1.00	1.64	0.33
66	0.19681	0.45838	5.84100	8.84950	0.19752	0.46264	5.68902	8.77968	0.36	0.93	2.60	0.79
67	0.04904	0.04904	0.75476	0.75476	0.04912	0.04912	0.77999	0.77999	0.16	0.16	3.34	3.34
68	0.08634	0.08634	1.34037	1.34037	0.08667	0.08667	1.36928	1.36928	0.38	0.38	2.16	2.16
69	0.15090	0.15090	2.49106	2.49106	0.15228	0.15228	2.48972	2.48972	0.91	0.91	0.05	0.05
70	0.18805	0.18805	3.37178	3.37178	0.18963	0.18963	3.28733	3.28733	0.84	0.84	2.50	2.50
71	0.19726	0.19726	3.67601	3.67601	0.19802	0.19802	3.52571	3.52571	0.39	0.39	4.09	4.09
72	0.19940	0.19940	3.76756	3.76756	0.19966	0.19966	3.58451	3.58451	0.13	0.13	4.86	4.86
73	0.34386	0.42911	0.50876	0.69349	0.34583	0.43021	0.52050	0.71300	0.57	0.26	2.31	2.81
74	0.45303	0.57303	1.02621	1.40056	0.45544	0.57370	1.03415	1.42270	0.53	0.12	0.77	1.58
75	0.55518	0.69135	2.61409	3.59731	0.55627	0.69138	2.61988	3.62230	0.20	0.00	0.22	0.69
76	0.59817	0.71971	5.25890	7.34597	0.59836	0.71978	5.26819	7.37190	0.03	0.01	0.18	0.35
77	0.61441	0.72052	7.87042	11.1220	0.61445	0.72054	7.88143	11.1484	0.01	0.00	0.14	0.24
78	0.62357	0.71695	10.4609	14.9062	0.62358	0.71696	10.4724	14.9330	0.00	0.00	0.11	0.18
79	0.24772	0.48061	0.38069	0.65670	0.24935	0.48204	0.38719	0.67577	0.66	0.30	1.71	2.90
80	0.31441	0.69337	0.73463	1.26883	0.31656	0.69496	0.73293	1.28843	0.69	0.23	0.23	1.54
81	0.36150	0.95207	1.77220	2.99415	0.36261	0.95382	1.75763	3.01124	0.31	0.18	0.82	0.57
82	0.37567	1.08243	3.47902	5.71377	0.37597	1.08376	3.46537	5.72869	0.08	0.12	0.39	0.26
83	0.38168	1.12893	5.15797	8.38634	0.38179	1.12974	5.14875	8.40082	0.03	0.07	0.18	0.17
84	0.38558	1.15129	6.82880	11.0752	0.38563	1.15178	6.82286	11.0901	0.01	0.04	0.09	0.13
85	0.08613	0.55997	0.13868	0.60001	0.08647	0.56101	0.13710	0.61809	0.40	0.19	1.14	3.01
86	0.09675	0.85438	0.22754	1.08582	0.09698	0.85604	0.21892	1.09990	0.24	0.19	3.79	1.30
87	0.09972	1.20704	0.46235	2.24564	0.09975	1.20885	0.44011	2.24730	0.03	0.15	4.81	0.07
88	0.09997	1.35165	0.75065	3.48150	0.09997	1.35267	0.71890	3.47037	0.00	0.08	4.23	0.32
89	0.09999	1.38616	0.89634	4.09073	0.09999	1.38661	0.86053	4.06839	0.00	0.03	3.99	0.55
90	0.09999	1.39584	0.95776	4.35465	0.09999	1.39601	0.91994	4.32452	0.00	0.01	3.95	0.69
91	0.25526	0.36418	0.36185	0.54477	0.25732	0.36565	0.36611	0.56224	0.81	0.41	1.18	3.21
92	0.33008	0.50566	0.65098	1.01370	0.33285	0.50772	0.64315	1.02885	0.84	0.41	1.20	1.49
93	0.38683	0.66361	1.22864	2.10224	0.38813	0.66641	1.19181	2.10098	0.34	0.42	3.00	0.06
94	0.39860	0.74567	1.69340	3.20528	0.39877	0.74780	1.63423	3.17053	0.04	0.28	3.49	1.08
95	0.39980	0.77594	1.92309	3.82361	0.39982	0.77713	1.85510	3.76408	0.01	0.15	3.54	1.56
96	0.39997	0.78895	2.05269	4.17840	0.39997	0.78958	1.97947	4.10218	0.00	0.08	3.57	1.82

Table B.7: Exhaustive case: Exact and approximation *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
97	0.08719	0.41566	0.12806	0.48041	0.08759	0.41663	0.12556	0.49564	0.46	0.23	1.95	3.17
98	0.09777	0.60091	0.18410	0.80068	0.09802	0.60236	0.17409	0.80919	0.25	0.24	5.44	1.06
99	0.09997	0.76875	0.25035	1.27825	0.09998	0.76971	0.22901	1.27182	0.01	0.12	8.52	0.50
100	0.10000	0.79837	0.27130	1.44351	0.10000	0.79850	0.24637	1.42902	0.00	0.02	9.19	1.00
101	0.10000	0.79989	0.27306	1.45875	0.10000	0.79991	0.24780	1.44285	0.00	0.00	9.25	1.09
102	0.10000	0.79999	0.27320	1.46009	0.10000	0.79999	0.24791	1.44401	0.00	0.00	9.26	1.10
103	0.08869	0.15949	0.11306	0.20253	0.08897	0.15976	0.11145	0.20534	0.32	0.17	1.42	1.39
104	0.09863	0.19142	0.13993	0.26346	0.09875	0.19160	0.13478	0.26210	0.12	0.09	3.68	0.52
105	0.09999	0.19991	0.14657	0.28555	0.09999	0.19992	0.13998	0.28214	0.00	0.00	4.50	1.19
106	0.10000	0.20000	0.14662	0.28592	0.10000	0.19999	0.14001	0.28246	0.00	0.00	4.50	1.21
107	0.10000	0.20000	0.14662	0.28592	0.10000	0.19999	0.14001	0.28246	0.00	0.00	4.50	1.21
108	0.10000	0.20000	0.14662	0.28592	0.10000	0.19999	0.14001	0.28246	0.00	0.00	4.50	1.21
109	0.09527	0.09673	0.86389	0.93090	0.09528	0.09675	0.87934	0.93998	0.01	0.02	1.79	0.98
110	0.18324	0.18700	1.62248	1.80796	0.18327	0.18706	1.64507	1.82260	0.02	0.03	1.39	0.81
111	0.35833	0.36903	3.57792	4.27519	0.35838	0.36918	3.60469	4.29669	0.01	0.04	0.75	0.50
112	0.48211	0.50001	6.46697	8.19993	0.48213	0.50014	6.49126	8.22432	0.01	0.02	0.38	0.30
113	0.53464	0.55610	9.20137	12.0494	0.53465	0.55617	9.22335	12.0748	0.00	0.01	0.24	0.21
114	0.56313	0.58626	11.8693	15.8686	0.56314	0.58630	11.8896	15.8947	0.00	0.01	0.17	0.16
115	0.07736	0.09860	0.80658	0.92957	0.07737	0.09863	0.82816	0.93877	0.02	0.03	2.67	0.99
116	0.13964	0.19676	1.47974	1.79789	0.13968	0.19689	1.50877	1.81302	0.03	0.07	1.96	0.84
117	0.24847	0.43114	3.10334	4.15227	0.24859	0.43162	3.13209	4.17556	0.05	0.11	0.93	0.56
118	0.31781	0.65638	5.33818	7.63023	0.31800	0.65704	5.35796	7.65779	0.06	0.10	0.37	0.36
119	0.34611	0.77863	7.37007	10.8508	0.34631	0.77924	7.38363	10.8808	0.06	0.08	0.18	0.28
120	0.36124	0.85437	9.30369	13.9470	0.36141	0.85488	9.31339	13.9792	0.05	0.06	0.10	0.23
121	0.04584	0.10180	0.54159	0.92728	0.04584	0.10186	0.56296	0.93667	0.02	0.06	3.94	1.01
122	0.07062	0.21168	0.85871	1.78244	0.07064	0.21193	0.87538	1.79810	0.03	0.12	1.94	0.88
123	0.09482	0.51294	1.33170	3.98641	0.09486	0.51407	1.33154	4.00974	0.05	0.22	0.01	0.59
124	0.09957	0.84462	1.80104	6.89403	0.09958	0.84703	1.78186	6.91519	0.02	0.28	1.06	0.31
125	0.09994	1.03241	2.22117	9.28738	0.09995	1.03551	2.18014	9.30242	0.01	0.30	1.85	0.16
126	0.09999	1.14671	2.61045	11.3626	0.09999	1.15010	2.54401	11.3705	0.00	0.30	2.55	0.07
127	0.07789	0.08124	0.80527	0.89844	0.07792	0.08129	0.82681	0.91309	0.04	0.06	2.67	1.63
128	0.14214	0.15182	1.46990	1.71971	0.14225	0.15203	1.49850	1.74272	0.08	0.14	1.95	1.34
129	0.26164	0.30054	2.97891	3.89075	0.26215	0.30160	3.00264	3.92260	0.19	0.35	0.80	0.82
130	0.34392	0.44315	4.71546	6.93496	0.34501	0.44558	4.71321	6.96497	0.32	0.55	0.05	0.43
131	0.37521	0.53316	5.98257	9.50813	0.37649	0.53669	5.94563	9.53125	0.34	0.66	0.62	0.24
132	0.38855	0.59756	6.98074	11.7205	0.38971	0.60210	6.89983	11.7318	0.30	0.76	1.16	0.10
133	0.04599	0.08373	0.54008	0.89532	0.04600	0.08383	0.56126	0.91014	0.04	0.11	3.92	1.65
134	0.07108	0.16270	0.84904	1.69915	0.07113	0.16308	0.86478	1.72241	0.07	0.23	1.85	1.37
135	0.09557	0.35419	1.23942	3.67418	0.09566	0.35591	1.23266	3.70266	0.10	0.49	0.55	0.78
136	0.09980	0.55335	1.46337	5.97628	0.09982	0.55662	1.41960	5.98596	0.02	0.59	2.99	0.16
137	0.09999	0.66225	1.61300	7.52307	0.09999	0.66564	1.52686	7.50432	0.00	0.51	5.34	0.25
138	0.10000	0.72268	1.73259	8.58364	0.10000	0.72547	1.60515	8.53469	0.00	0.39	7.36	0.57
139	0.04613	0.05776	0.53866	0.71116	0.04621	0.05789	0.55907	0.73697	0.18	0.21	3.79	3.63
140	0.07139	0.09971	0.84175	1.22419	0.07162	0.10011	0.85384	1.25052	0.33	0.40	1.44	2.15
141	0.09579	0.16652	1.18818	2.05338	0.09608	0.16723	1.16416	2.05259	0.30	0.43	2.02	0.04
142	0.09983	0.19519	1.28749	2.43488	0.09987	0.19542	1.23145	2.40744	0.04	0.12	4.35	1.13
143	0.09999	0.19934	1.30043	2.49882	0.09999	0.19938	1.23630	2.46546	0.00	0.02	4.93	1.34
144	0.10000	0.19991	1.30252	2.50895	0.10000	0.19991	1.23680	2.47457	0.00	0.00	5.05	1.37

Table B.8: Exhaustive case: Exact and approximation *(continued)*

Appendix C

Gated Case

NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2	NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S_1	S_2
1	0.8	1.0	0.8	1.0	1.0	1.0	1	1	49	0.7	1.0	1.4	2.0	1.0	2.0	1	1
2	0.8	1.0	0.8	1.0	1.0	1.0	2	2	50	0.7	1.0	1.4	2.0	1.0	2.0	2	2
3	0.8	1.0	0.8	1.0	1.0	1.0	5	5	51	0.7	1.0	1.4	2.0	1.0	2.0	5	5
4	0.8	1.0	0.8	1.0	1.0	1.0	10	10	52	0.7	1.0	1.4	2.0	1.0	2.0	10	10
5	0.5	1.0	0.8	1.0	1.0	1.0	1	1	53	0.4	1.0	1.4	2.0	1.0	2.0	1	1
6	0.5	1.0	0.8	1.0	1.0	1.0	2	2	54	0.4	1.0	1.4	2.0	1.0	2.0	2	2
7	0.5	1.0	0.8	1.0	1.0	1.0	5	5	55	0.4	1.0	1.4	2.0	1.0	2.0	5	5
8	0.5	1.0	0.8	1.0	1.0	1.0	10	10	56	0.4	1.0	1.4	2.0	1.0	2.0	10	10
9	0.2	1.0	0.8	1.0	1.0	1.0	1	1	57	0.1	1.0	1.4	2.0	1.0	2.0	1	1
10	0.2	1.0	0.8	1.0	1.0	1.0	2	2	58	0.1	1.0	1.4	2.0	1.0	2.0	2	2
11	0.2	1.0	0.8	1.0	1.0	1.0	5	5	59	0.1	1.0	1.4	2.0	1.0	2.0	5	5
12	0.2	1.0	0.8	1.0	1.0	1.0	10	10	60	0.1	1.0	1.4	2.0	1.0	2.0	10	10
13	0.5	1.0	0.5	1.0	1.0	1.0	1	1	61	0.4	1.0	0.8	2.0	1.0	2.0	1	1
14	0.5	1.0	0.5	1.0	1.0	1.0	2	2	62	0.4	1.0	0.8	2.0	1.0	2.0	2	2
15	0.5	1.0	0.5	1.0	1.0	1.0	5	5	63	0.4	1.0	0.8	2.0	1.0	2.0	5	5
16	0.5	1.0	0.5	1.0	1.0	1.0	10	10	64	0.4	1.0	0.8	2.0	1.0	2.0	10	10
17	0.2	1.0	0.5	1.0	1.0	1.0	1	1	65	0.1	1.0	0.8	2.0	1.0	2.0	1	1
18	0.2	1.0	0.5	1.0	1.0	1.0	2	2	66	0.1	1.0	0.8	2.0	1.0	2.0	2	2
19	0.2	1.0	0.5	1.0	1.0	1.0	5	5	67	0.1	1.0	0.8	2.0	1.0	2.0	5	5
20	0.2	1.0	0.5	1.0	1.0	1.0	10	10	68	0.1	1.0	0.8	2.0	1.0	2.0	10	10
21	0.2	1.0	0.2	1.0	1.0	1.0	1	1	69	0.1	1.0	0.2	2.0	1.0	2.0	1	1
22	0.2	1.0	0.2	1.0	1.0	1.0	2	2	70	0.1	1.0	0.2	2.0	1.0	2.0	2	2
23	0.2	1.0	0.2	1.0	1.0	1.0	5	5	71	0.1	1.0	0.2	2.0	1.0	2.0	5	5
24	0.2	1.0	0.2	1.0	1.0	1.0	10	10	72	0.1	1.0	0.2	2.0	1.0	2.0	10	10
25	0.8	1.0	0.8	1.0	0.1	0.1	1	1	73	0.7	1.0	1.4	2.0	0.1	0.2	1	1
26	0.8	1.0	0.8	1.0	0.1	0.1	2	2	74	0.7	1.0	1.4	2.0	0.1	0.2	2	2
27	0.8	1.0	0.8	1.0	0.1	0.1	5	5	75	0.7	1.0	1.4	2.0	0.1	0.2	5	5
28	0.8	1.0	0.8	1.0	0.1	0.1	10	10	76	0.7	1.0	1.4	2.0	0.1	0.2	10	10
29	0.5	1.0	0.8	1.0	0.1	0.1	1	1	77	0.4	1.0	1.4	2.0	0.1	0.2	1	1
30	0.5	1.0	0.8	1.0	0.1	0.1	2	2	78	0.4	1.0	1.4	2.0	0.1	0.2	2	2
31	0.5	1.0	0.8	1.0	0.1	0.1	5	5	79	0.4	1.0	1.4	2.0	0.1	0.2	5	5
32	0.5	1.0	0.8	1.0	0.1	0.1	10	10	80	0.4	1.0	1.4	2.0	0.1	0.2	10	10
33	0.2	1.0	0.8	1.0	0.1	0.1	1	1	81	0.1	1.0	1.4	2.0	0.1	0.2	1	1
34	0.2	1.0	0.8	1.0	0.1	0.1	2	2	82	0.1	1.0	1.4	2.0	0.1	0.2	2	2
35	0.2	1.0	0.8	1.0	0.1	0.1	5	5	83	0.1	1.0	1.4	2.0	0.1	0.2	5	5
36	0.2	1.0	0.8	1.0	0.1	0.1	10	10	84	0.1	1.0	1.4	2.0	0.1	0.2	10	10
37	0.5	1.0	0.5	1.0	0.1	0.1	1	1	85	0.4	1.0	0.8	2.0	0.1	0.2	1	1
38	0.5	1.0	0.5	1.0	0.1	0.1	2	2	86	0.4	1.0	0.8	2.0	0.1	0.2	2	2
39	0.5	1.0	0.5	1.0	0.1	0.1	5	5	87	0.4	1.0	0.8	2.0	0.1	0.2	5	5
40	0.5	1.0	0.5	1.0	0.1	0.1	10	10	88	0.4	1.0	0.8	2.0	0.1	0.2	10	10
41	0.2	1.0	0.5	1.0	0.1	0.1	1	1	89	0.1	1.0	0.8	2.0	0.1	0.2	1	1
42	0.2	1.0	0.5	1.0	0.1	0.1	2	2	90	0.1	1.0	0.8	2.0	0.1	0.2	2	2
43	0.2	1.0	0.5	1.0	0.1	0.1	5	5	91	0.1	1.0	0.8	2.0	0.1	0.2	5	5
44	0.2	1.0	0.5	1.0	0.1	0.1	10	10	92	0.1	1.0	0.8	2.0	0.1	0.2	10	10
45	0.2	1.0	0.2	1.0	0.1	0.1	1	1	93	0.1	1.0	0.2	2.0	0.1	0.2	1	1
46	0.2	1.0	0.2	1.0	0.1	0.1	2	2	94	0.1	1.0	0.2	2.0	0.1	0.2	2	2
47	0.2	1.0	0.2	1.0	0.1	0.1	5	5	95	0.1	1.0	0.2	2.0	0.1	0.2	5	5
48	0.2	1	0.2	1.0	0.1	0.1	10	10	96	0.1	1.0	0.2	2.0	0.1	0.2	10	10

Table C.1: Gated Case: Problem data

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.23395	0.23395	0.70756	0.70756	0.23498	0.23498	0.70626	0.706269	0.44	0.44	0.18	0.18
2	0.32099	0.32099	1.46974	1.46974	0.32175	0.32175	1.46942	1.469421	0.24	0.24	0.02	0.02
3	0.41256	0.41256	3.94933	3.94933	0.41212	0.41212	3.95577	3.955770	0.10	0.10	0.16	0.16
4	0.45365	0.45365	8.35154	8.35154	0.45312	0.45312	8.35899	8.358998	0.12	0.12	0.09	0.09
5	0.20914	0.23888	0.58172	0.70139	0.21025	0.24047	0.57948	0.699406	0.53	0.67	0.38	0.28
6	0.28720	0.33178	1.17955	1.44819	0.28848	0.33362	1.17685	1.446737	0.44	0.55	0.23	0.10
7	0.37309	0.43445	3.06375	3.86200	0.37369	0.43472	3.07325	3.876174	0.16	0.06	0.31	0.37
8	0.41660	0.48158	6.39016	8.17957	0.41669	0.48040	6.42384	8.218251	0.02	0.25	0.53	0.47
9	0.13539	0.25479	0.32304	0.68150	0.13601	0.25676	0.31994	0.679038	0.46	0.77	0.96	0.36
10	0.17292	0.36988	0.56519	1.37346	0.17360	0.37348	0.55810	1.369286	0.39	0.97	1.26	0.30
11	0.19685	0.52709	1.07794	3.47119	0.19712	0.53287	1.06602	3.476951	0.14	1.10	1.11	0.17
12	0.19985	0.62822	1.72312	6.98045	0.19988	0.63432	1.72432	7.053401	0.01	0.97	0.07	1.04
13	0.21256	0.21256	0.57486	0.57486	0.21428	0.21428	0.57142	0.571429	0.81	0.81	0.60	0.60
14	0.29435	0.29435	1.15219	1.15219	0.29705	0.29705	1.14578	1.145779	0.92	0.92	0.56	0.56
15	0.38696	0.38696	2.89806	2.89806	0.39054	0.39054	2.90170	2.901701	0.93	0.93	0.13	0.13
16	0.43472	0.43472	5.80020	5.80020	0.43848	0.43848	5.88464	5.884643	0.86	0.86	1.46	1.46
17	0.13651	0.22373	0.31740	0.55253	0.13744	0.22586	0.31279	0.548266	0.68	0.95	1.45	0.77
18	0.17448	0.31899	0.54168	1.06011	0.17569	0.32338	0.52856	1.048235	0.70	1.38	2.42	1.12
19	0.19765	0.43497	0.92816	2.29026	0.19824	0.44320	0.87708	2.230160	0.30	1.89	5.50	2.62
20	0.19994	0.48670	1.18408	3.42946	0.19998	0.49244	1.05450	3.162580	0.02	1.18	10.9	7.78
21	0.14015	0.14015	0.29922	0.29922	0.14124	0.14124	0.29378	0.293787	0.78	0.78	1.82	1.82
22	0.17932	0.17932	0.47118	0.47118	0.18077	0.18077	0.45469	0.454699	0.81	0.81	3.50	3.50
23	0.19917	0.19917	0.62167	0.62167	0.19946	0.19946	0.57663	0.576638	0.15	0.15	7.24	7.24
24	0.19999	0.19999	0.63326	0.63326	0.19999	0.19999	0.58259	0.582599	0.00	0.00	8.00	8.00
25	0.04529	0.04529	0.94338	0.94338	0.04529	0.04529	0.94338	0.943380	0.00	0.00	0.00	0.00
26	0.08308	0.08308	1.86712	1.86712	0.08308	0.08308	1.86711	1.867115	0.00	0.00	0.00	0.00
27	0.16638	0.16638	4.59548	4.59548	0.16639	0.16639	4.59550	4.595509	0.00	0.00	0.00	0.00
28	0.24983	0.24983	9.11786	9.11786	0.24982	0.24982	9.11794	9.117946	0.00	0.00	0.00	0.00
29	0.04491	0.04530	0.91016	0.94336	0.04492	0.04531	0.91015	0.943359	0.01	0.01	0.00	0.00
30	0.08224	0.08314	1.77968	1.86701	0.08225	0.08315	1.77967	1.867002	0.01	0.02	0.00	0.00
31	0.16426	0.16675	4.26012	4.59444	0.16427	0.16677	4.26015	4.594530	0.01	0.01	0.00	0.00
32	0.24626	0.25094	8.19736	9.11334	0.24627	0.25091	8.19768	9.113921	0.00	0.01	0.00	0.01
33	0.04204	0.04542	0.78978	0.94322	0.04204	0.04544	0.78976	0.943200	0.01	0.04	0.00	0.00
34	0.07472	0.08373	1.45704	1.86601	0.07474	0.08379	1.45695	1.865958	0.02	0.07	0.01	0.00
35	0.13705	0.17173	2.98501	4.58116	0.13708	0.17193	2.98460	4.581307	0.02	0.12	0.01	0.00
36	0.18045	0.27169	4.61818	9.03139	0.18048	0.27209	4.61713	9.033529	0.02	0.15	0.02	0.02
37	0.04492	0.04492	0.91014	0.91014	0.04493	0.04493	0.91012	0.910125	0.02	0.02	0.00	0.00
38	0.08229	0.08229	1.77954	1.77954	0.08232	0.08232	1.77949	1.779493	0.03	0.03	0.00	0.00
39	0.16453	0.16453	4.25833	4.25833	0.16461	0.16461	4.25838	4.258381	0.05	0.05	0.00	0.00
40	0.24710	0.24710	8.18681	8.18681	0.24724	0.24724	8.18869	8.188697	0.06	0.06	0.02	0.02
41	0.04204	0.04502	0.78977	0.90994	0.04205	0.04505	0.78971	0.909890	0.03	0.07	0.01	0.01
42	0.07473	0.08278	1.45691	1.77810	0.07478	0.08289	1.45662	1.777873	0.06	0.14	0.02	0.01
43	0.13709	0.16855	2.98260	4.23532	0.13725	0.16914	2.98051	4.234810	0.12	0.35	0.07	0.01
44	0.18051	0.26344	4.59769	8.00947	0.18079	0.26510	4.58753	8.014055	0.15	0.63	0.22	0.06
45	0.04209	0.04209	0.78954	0.78954	0.04214	0.04214	0.78930	0.789301	0.11	0.11	0.03	0.03
46	0.07493	0.07493	1.45502	1.45502	0.07513	0.07513	1.45372	1.453723	0.26	0.26	0.09	0.09
47	0.13814	0.13814	2.94841	2.94841	0.13911	0.13911	2.93567	2.935672	0.70	0.70	0.43	0.43
48	0.18239	0.18239	4.33106	4.33106	0.18400	0.18400	4.25855	4.258559	0.88	0.88	1.67	1.67

Table C.2: Gated Case: Exact and decomposed

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
49	0.28187	0.33399	0.59732	0.76143	0.28319	0.33541	0.59544	0.760421	0.47	0.42	0.32	0.13
50	0.38695	0.46596	1.23048	1.55872	0.38861	0.46597	1.22759	1.559725	0.43	0.00	0.23	0.06
51	0.50413	0.60796	3.27856	4.08830	0.50513	0.60338	3.28022	4.102784	0.20	0.75	0.05	0.35
52	0.56517	0.66580	6.96696	8.51924	0.56551	0.65938	6.97624	8.545674	0.06	0.96	0.13	0.31
53	0.22742	0.35338	0.43142	0.74758	0.22853	0.35530	0.42866	0.746211	0.49	0.54	0.64	0.18
54	0.30387	0.51487	0.82357	1.50701	0.30543	0.51712	0.81640	1.506182	0.51	0.44	0.87	0.05
55	0.37406	0.74010	1.83229	3.82403	0.37531	0.74364	1.81201	3.834413	0.33	0.48	1.11	0.27
56	0.39544	0.89830	3.17554	7.74301	0.39588	0.90364	3.14651	7.797634	0.11	0.59	0.91	0.71
57	0.08663	0.40499	0.13361	0.71072	0.08682	0.40635	0.13171	0.709747	0.22	0.34	1.42	0.14
58	0.09785	0.63458	0.18782	1.37973	0.09795	0.63756	0.18322	1.377554	0.10	0.47	2.45	0.16
59	0.09998	0.98424	0.26899	3.25691	0.09999	0.99098	0.25965	3.254211	0.00	0.68	3.47	0.08
60	0.10000	1.21038	0.37076	6.00081	0.10000	1.22110	0.35760	5.999987	0.00	0.89	3.55	0.01
61	0.22872	0.31137	0.42818	0.61077	0.23061	0.31363	0.42347	0.607959	0.82	0.72	1.10	0.46
62	0.30613	0.44789	0.80813	1.18458	0.30931	0.45204	0.79370	1.178422	1.04	0.93	1.79	0.52
63	0.37678	0.62668	1.70685	2.70525	0.38041	0.63886	1.63648	2.673225	0.96	1.94	4.12	1.18
64	0.39684	0.73378	2.60664	4.55582	0.39843	0.75425	2.35458	4.338724	0.40	2.79	9.67	4.77
65	0.08676	0.34718	0.13233	0.56602	0.08707	0.34877	0.12924	0.564036	0.36	0.46	2.34	0.35
66	0.09793	0.52204	0.18218	1.02098	0.09811	0.52558	0.17413	1.014729	0.18	0.68	4.42	0.61
67	0.09999	0.73023	0.23443	1.91202	0.09999	0.73597	0.21519	1.878835	0.00	0.79	8.21	1.74
68	0.10000	0.79367	0.25918	2.41469	0.10000	0.79566	0.23080	2.314413	0.00	0.25	10.9	4.15
69	0.08756	0.15593	0.12438	0.22031	0.08788	0.15635	0.12119	0.218205	0.36	0.27	2.56	0.96
70	0.09834	0.19013	0.15704	0.29346	0.09849	0.19048	0.15047	0.288634	0.15	0.18	4.18	1.65
71	0.09999	0.19990	0.16597	0.32165	0.09999	0.19992	0.15734	0.314420	0.00	0.01	5.20	2.25
72	0.10000	0.20000	0.16604	0.32207	0.10000	0.20000	0.15737	0.314753	0.00	0.00	5.22	2.27
73	0.05991	0.06052	0.91441	0.95676	0.05991	0.06052	0.91440	0.956765	0.01	0.01	0.00	0.00
74	0.10971	0.11110	1.79618	1.89723	0.10972	0.11111	1.79617	1.897234	0.01	0.00	0.00	0.00
75	0.21925	0.22290	4.35702	4.67781	0.21926	0.22283	4.35702	4.677953	0.00	0.03	0.00	0.00
76	0.32910	0.33536	8.53436	9.27211	0.32911	0.33510	8.53441	9.272776	0.00	0.08	0.00	0.01
77	0.05830	0.06061	0.85424	0.95670	0.05830	0.06062	0.85423	0.956695	0.01	0.02	0.00	0.00
78	0.10567	0.11157	1.63608	1.89679	0.10568	0.11158	1.63605	1.896785	0.01	0.02	0.00	0.00
79	0.20563	0.22662	3.71826	4.67219	0.20565	0.22655	3.71818	4.672457	0.01	0.03	0.00	0.01
80	0.29704	0.35106	6.64340	9.23675	0.29706	0.35069	6.64326	9.238216	0.01	0.11	0.00	0.02
81	0.04337	0.06152	0.56624	0.95605	0.04337	0.06154	0.56622	0.956040	0.01	0.04	0.00	0.00
82	0.06855	0.11596	0.91389	1.89265	0.06856	0.11603	0.91377	1.892611	0.01	0.06	0.01	0.00
83	0.09510	0.25684	1.35466	4.62667	0.09511	0.25711	1.35384	4.626647	0.01	0.11	0.06	0.00
84	0.09981	0.44549	1.54827	9.01629	0.09981	0.44615	1.54558	9.017191	0.00	0.15	0.17	0.01
85	0.05830	0.06019	0.85424	0.92475	0.05831	0.06021	0.85420	0.924729	0.02	0.03	0.00	0.00
86	0.10567	0.11065	1.63605	1.81169	0.10571	0.11072	1.63592	1.811621	0.04	0.06	0.01	0.00
87	0.20564	0.22422	3.71751	4.33718	0.20580	0.22441	3.71687	4.337409	0.08	0.09	0.02	0.01
88	0.29707	0.34612	6.63680	8.28073	0.29742	0.34654	6.63486	8.284781	0.12	0.12	0.03	0.05
89	0.04337	0.06101	0.56627	0.92373	0.04338	0.06106	0.56618	0.923672	0.02	0.08	0.02	0.01
90	0.06854	0.11457	0.91403	1.80473	0.06857	0.11475	0.91357	1.804506	0.03	0.15	0.05	0.01
91	0.09509	0.25021	1.35549	4.24981	0.09512	0.25104	1.35160	4.248804	0.03	0.33	0.29	0.02
92	0.09980	0.42180	1.54849	7.78850	0.09981	0.42397	1.53269	7.785946	0.01	0.51	1.02	0.03
93	0.04337	0.05347	0.56629	0.73260	0.04343	0.05357	0.56564	0.732111	0.15	0.19	0.12	0.07
94	0.06853	0.09344	0.91367	1.29233	0.06872	0.09378	0.91022	1.289900	0.29	0.36	0.38	0.19
95	0.09505	0.16173	1.34367	2.28049	0.09532	0.16250	1.31956	2.264263	0.29	0.48	1.79	0.71
96	0.09979	0.19419	1.46898	2.78620	0.09984	0.19450	1.41489	2.747523	0.04	0.16	3.68	1.39

Table C.3: Gated Case: Exact and decomposed *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.23395	0.23395	0.70756	0.70756	0.23498	0.23498	0.70626	0.70626	0.44	0.44	0.18	0.18
2	0.32099	0.32099	1.46974	1.46974	0.32314	0.32314	1.53711	1.53711	0.67	0.67	4.58	4.58
3	0.41256	0.41256	3.94933	3.94933	0.41370	0.41370	4.22331	4.22331	0.28	0.28	6.94	6.94
4	0.45365	0.45365	8.35154	8.35154	0.45390	0.45390	8.86783	8.86783	0.06	0.06	6.18	6.18
5	0.20914	0.23888	0.58172	0.70139	0.21025	0.24047	0.57947	0.69939	0.53	0.67	0.39	0.28
6	0.28720	0.33178	1.17955	1.44819	0.29016	0.33537	1.21732	1.51625	1.03	1.08	3.20	4.70
7	0.37309	0.43445	3.06375	3.86200	0.37622	0.43724	3.20277	4.15417	0.84	0.64	4.54	7.57
8	0.41660	0.48158	6.39016	8.17957	0.41870	0.48197	6.62295	8.74977	0.51	0.08	3.64	6.97
9	0.13539	0.25479	0.32304	0.68150	0.13601	0.25676	0.31995	0.67903	0.46	0.77	0.96	0.36
10	0.17292	0.36988	0.56519	1.37346	0.17418	0.37644	0.56443	1.44471	0.73	1.78	0.14	5.19
11	0.19685	0.52709	1.07794	3.47119	0.19733	0.53980	1.06976	3.79150	0.24	2.41	0.76	9.23
12	0.19985	0.62822	1.72312	6.98045	0.19989	0.64168	1.72467	7.65683	0.02	2.14	0.09	9.69
13	0.21256	0.21256	0.57486	0.57486	0.21428	0.21428	0.57142	0.57142	0.81	0.81	0.60	0.60
14	0.29435	0.29435	1.15219	1.15219	0.29902	0.29903	1.18650	1.18650	1.59	1.59	2.98	2.98
15	0.38696	0.38696	2.89806	2.89806	0.39383	0.39383	3.02733	3.02733	1.77	1.78	4.46	4.46
16	0.43472	0.43472	5.80020	5.80020	0.44114	0.44114	6.06216	6.06217	1.48	1.48	4.52	4.52
17	0.13651	0.22373	0.31740	0.55253	0.13744	0.22586	0.31278	0.54826	0.68	0.95	1.46	0.77
18	0.17448	0.31899	0.54168	1.06011	0.17630	0.32617	0.53452	1.08957	1.04	2.25	1.32	2.78
19	0.19765	0.43497	0.92816	2.29026	0.19841	0.44792	0.87953	2.33154	0.38	2.98	5.24	1.80
20	0.19994	0.48670	1.18408	3.42946	0.19998	0.49409	1.05456	3.21400	0.02	1.52	10.9	6.28
21	0.14015	0.14015	0.29922	0.29922	0.14124	0.14124	0.29378	0.29378	0.78	0.78	1.82	1.82
22	0.17932	0.17932	0.47118	0.47118	0.18141	0.18140	0.45987	0.45987	1.16	1.16	2.40	2.40
23	0.19917	0.19917	0.62167	0.62167	0.19955	0.19955	0.57760	0.57760	0.19	0.19	7.09	7.09
24	0.19999	0.19999	0.63326	0.63326	0.20000	0.20000	0.58260	0.58260	0.00	0.00	8.00	8.00
25	0.04529	0.04529	0.94338	0.94338	0.04529	0.04529	0.94334	0.94334	0.00	0.00	0.00	0.00
26	0.08308	0.08308	1.86712	1.86712	0.08310	0.08310	1.88544	1.88544	0.02	0.02	0.98	0.98
27	0.16638	0.16638	4.59548	4.59548	0.16643	0.16643	4.70746	4.70746	0.03	0.03	2.44	2.44
28	0.24983	0.24983	9.11786	9.11786	0.24986	0.24986	9.40420	9.40420	0.01	0.01	3.14	3.14
29	0.04491	0.04530	0.91016	0.94336	0.04492	0.04531	0.91012	0.94332	0.01	0.01	0.00	0.00
30	0.08224	0.08314	1.77968	1.86701	0.08229	0.08317	1.79317	1.88536	0.05	0.04	0.76	0.98
31	0.16426	0.16675	4.26012	4.59444	0.16442	0.16682	4.33101	4.70675	0.10	0.04	1.66	2.44
32	0.24626	0.25094	8.19736	9.11334	0.24652	0.25095	8.35099	9.40141	0.11	0.01	1.87	3.16
33	0.04204	0.04542	0.78978	0.94322	0.04204	0.04544	0.78976	0.94313	0.01	0.04	0.00	0.01
34	0.07472	0.08373	1.45704	1.86601	0.07480	0.08381	1.46253	1.88443	0.10	0.09	0.38	0.99
35	0.13705	0.17173	2.98501	4.58116	0.13730	0.17202	3.00096	4.69679	0.18	0.17	0.53	2.52
36	0.18045	0.27169	4.61818	9.03139	0.18068	0.27229	4.63093	9.34394	0.13	0.22	0.28	3.46
37	0.04492	0.04492	0.91014	0.91014	0.04493	0.04493	0.91010	0.91010	0.02	0.02	0.00	0.00
38	0.08229	0.08229	1.77954	1.77954	0.08235	0.08235	1.79302	1.79302	0.08	0.08	0.76	0.76
39	0.16453	0.16453	4.25833	4.25833	0.16477	0.16477	4.32938	4.32938	0.15	0.15	1.67	1.67
40	0.24710	0.24710	8.18681	8.18681	0.24751	0.24751	8.34231	8.34231	0.17	0.17	1.90	1.90
41	0.04204	0.04502	0.78977	0.90994	0.04205	0.04505	0.78970	0.90987	0.03	0.07	0.01	0.01
42	0.07473	0.08278	1.45691	1.77810	0.07484	0.08293	1.46222	1.79151	0.14	0.19	0.36	0.75
43	0.13709	0.16855	2.98260	4.23532	0.13748	0.16935	2.99688	4.30741	0.28	0.48	0.48	1.70
44	0.18051	0.26344	4.59769	8.00947	0.18099	0.26566	4.60102	8.17382	0.26	0.84	0.07	2.05
45	0.04209	0.04209	0.78954	0.78954	0.04214	0.04214	0.78929	0.78929	0.11	0.11	0.03	0.03
46	0.07493	0.07493	1.45502	1.45502	0.07519	0.07519	1.45930	1.45930	0.35	0.35	0.29	0.29
47	0.13814	0.13814	2.94841	2.94841	0.13935	0.13935	2.95171	2.95171	0.87	0.87	0.11	0.11
48	0.18239	0.18239	4.33106	4.33106	0.18419	0.18419	4.26978	4.26978	0.99	0.99	1.41	1.41

Table C.4: Gated Case: Exact and approximation

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
49	0.28187	0.33399	0.59732	0.76143	0.28319	0.33541	0.59543	0.76041	0.47	0.42	0.32	0.13
50	0.38695	0.46596	1.23048	1.55872	0.39173	0.46740	1.29721	1.60566	1.24	0.31	5.42	3.01
51	0.50413	0.60796	3.27856	4.08830	0.51021	0.60508	3.53579	4.27915	1.21	0.47	7.85	4.67
52	0.56517	0.66580	6.96696	8.51924	0.56986	0.66013	7.42923	8.86434	0.83	0.85	6.64	4.05
53	0.22742	0.35338	0.43142	0.74758	0.22853	0.35530	0.42867	0.74619	0.49	0.54	0.64	0.19
54	0.30387	0.51487	0.82357	1.50701	0.30778	0.51958	0.84468	1.55609	1.28	0.91	2.56	3.26
55	0.37406	0.74010	1.83229	3.82403	0.37769	0.74934	1.86173	4.04089	0.97	1.25	1.61	5.67
56	0.39544	0.89830	3.17554	7.74301	0.39659	0.90967	3.17087	8.20001	0.29	1.27	0.15	5.90
57	0.08663	0.40499	0.13361	0.71072	0.08682	0.40635	0.13171	0.70974	0.22	0.34	1.42	0.14
58	0.09785	0.63458	0.18782	1.37973	0.09803	0.64239	0.18373	1.43643	0.18	1.23	2.18	4.11
59	0.09998	0.98424	0.26899	3.25691	0.09999	1.00475	0.25965	3.49717	0.00	2.08	3.47	7.38
60	0.10000	1.21038	0.37076	6.00081	0.09999	1.23576	0.35756	6.39443	0.00	2.10	3.56	6.56
61	0.22872	0.31137	0.42818	0.61077	0.23060	0.31363	0.42347	0.60794	0.82	0.72	1.10	0.46
62	0.30613	0.44789	0.80813	1.18458	0.31179	0.45449	0.82156	1.20446	1.85	1.47	1.66	1.68
63	0.37678	0.62668	1.70685	2.70525	0.38275	0.64366	1.67907	2.74345	1.58	2.71	1.63	1.41
64	0.39684	0.73378	2.60664	4.55582	0.39882	0.75734	2.36550	4.39764	0.50	3.21	9.25	3.47
65	0.08676	0.34718	0.13233	0.56602	0.08707	0.34877	0.12923	0.56403	0.36	0.46	2.34	0.35
66	0.09793	0.52204	0.18218	1.02098	0.09818	0.52939	0.17462	1.04186	0.26	1.41	4.15	2.05
67	0.09999	0.73023	0.23443	1.91202	0.09999	0.74140	0.21519	1.92847	0.00	1.53	8.21	0.86
68	0.10000	0.79367	0.25918	2.41469	0.10000	0.79658	0.23076	2.32630	0.00	0.37	10.9	3.66
69	0.08756	0.15593	0.12438	0.22031	0.08788	0.15635	0.12119	0.21818	0.36	0.27	2.56	0.97
70	0.09834	0.19013	0.15704	0.29346	0.09856	0.19067	0.15087	0.28939	0.22	0.28	3.93	1.38
71	0.09999	0.19990	0.16597	0.32165	0.09999	0.19992	0.15732	0.31445	0.00	0.01	5.22	2.24
72	0.10000	0.20000	0.16604	0.32207	0.10000	0.20000	0.15735	0.31474	0.00	0.00	5.24	2.28
73	0.05991	0.06052	0.91441	0.95676	0.05991	0.06052	0.91436	0.95666	0.01	0.01	0.01	0.01
74	0.10971	0.11110	1.79618	1.89723	0.10979	0.11111	1.81846	1.90849	0.07	0.01	1.24	0.59
75	0.21925	0.22290	4.35702	4.67781	0.21959	0.22285	4.48681	4.74476	0.16	0.02	2.98	1.43
76	0.32910	0.33536	8.53436	9.27211	0.32967	0.33511	8.84915	9.43687	0.17	0.08	3.69	1.78
77	0.05830	0.06061	0.85424	0.95670	0.05830	0.06062	0.85420	0.95658	0.01	0.02	0.00	0.01
78	0.10567	0.11157	1.63608	1.89679	0.10580	0.11159	1.65046	1.90806	0.13	0.02	0.88	0.59
79	0.20563	0.22662	3.71826	4.67219	0.20627	0.22657	3.78264	4.74042	0.31	0.02	1.73	1.46
80	0.29704	0.35106	6.64340	9.23675	0.29812	0.35073	6.75055	9.40973	0.36	0.10	1.61	1.87
81	0.04337	0.06152	0.56624	0.95605	0.04337	0.06154	0.56622	0.95596	0.00	0.04	0.00	0.01
82	0.06855	0.11596	0.91389	1.89265	0.06861	0.11605	0.91564	1.90435	0.09	0.08	0.19	0.62
83	0.09510	0.25684	1.35466	4.62667	0.09515	0.25728	1.35500	4.70338	0.05	0.17	0.03	1.66
84	0.09981	0.44549	1.54827	9.01629	0.09981	0.44673	1.54563	9.23270	0.00	0.28	0.17	2.40
85	0.05830	0.06019	0.85424	0.92475	0.05831	0.06021	0.85418	0.92470	0.02	0.03	0.01	0.01
86	0.10567	0.11065	1.63605	1.81169	0.10584	0.11074	1.65034	1.81936	0.16	0.08	0.87	0.42
87	0.20564	0.22422	3.71751	4.33718	0.20642	0.22451	3.78137	4.37632	0.38	0.13	1.72	0.90
88	0.29707	0.34612	6.63680	8.28073	0.29849	0.34674	6.74215	8.36637	0.48	0.18	1.59	1.03
89	0.04337	0.06101	0.56627	0.92373	0.04338	0.06106	0.56618	0.92361	0.02	0.08	0.02	0.01
90	0.06854	0.11457	0.91403	1.80473	0.06862	0.11479	0.91543	1.81249	0.12	0.19	0.15	0.43
91	0.09509	0.25021	1.35549	4.24981	0.09516	0.25136	1.35278	4.29129	0.08	0.46	0.20	0.98
92	0.09980	0.42180	1.54849	7.78850	0.09981	0.42497	1.53279	7.87624	0.01	0.75	1.01	1.13
93	0.04337	0.05347	0.56629	0.73260	0.04343	0.05357	0.56564	0.73210	0.15	0.19	0.11	0.07
94	0.06853	0.09344	0.91367	1.29233	0.06878	0.09382	0.91206	1.29159	0.37	0.40	0.18	0.06
95	0.09505	0.16173	1.34367	2.28049	0.09536	0.16260	1.32065	2.26595	0.33	0.54	1.71	0.59
96	0.09979	0.19419	1.46898	2.78620	0.09984	0.19452	1.41497	2.74829	0.05	0.17	3.68	1.36

Table C.5: Gated Case: Exact and approximation *(continued)*

Appendix D

G-Limited Case

Note: The instances that do not converge after 99 iterations are: **25, 43, 49, 52, 125, 143, 152.**

NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S	K	NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S	K
1	0.8	1.0	0.8	1.0	1.0	1.0	2	1	55	0.8	1.0	0.8	1.0	0.1	0.1	2	1
2	0.8	1.0	0.8	1.0	1.0	1.0	5	1	56	0.8	1.0	0.8	1.0	0.1	0.1	5	1
3	0.8	1.0	0.8	1.0	1.0	1.0	5	2	57	0.8	1.0	0.8	1.0	0.1	0.1	5	2
4	0.8	1.0	0.8	1.0	1.0	1.0	10	1	58	0.8	1.0	0.8	1.0	0.1	0.1	10	1
5	0.8	1.0	0.8	1.0	1.0	1.0	10	2	59	0.8	1.0	0.8	1.0	0.1	0.1	10	2
6	0.8	1.0	0.8	1.0	1.0	1.0	10	5	60	0.8	1.0	0.8	1.0	0.1	0.1	10	5
7	0.8	1.0	0.8	1.0	1.0	1.0	15	1	61	0.8	1.0	0.8	1.0	0.1	0.1	15	1
8	0.8	1.0	0.8	1.0	1.0	1.0	15	2	62	0.8	1.0	0.8	1.0	0.1	0.1	15	2
9	0.8	1.0	0.8	1.0	1.0	1.0	15	5	63	0.8	1.0	0.8	1.0	0.1	0.1	15	5
10	0.5	1.0	0.8	1.0	1.0	1.0	2	1	64	0.5	1.0	0.8	1.0	0.1	0.1	2	1
11	0.5	1.0	0.8	1.0	1.0	1.0	5	1	65	0.5	1.0	0.8	1.0	0.1	0.1	5	1
12	0.5	1.0	0.8	1.0	1.0	1.0	5	2	66	0.5	1.0	0.8	1.0	0.1	0.1	5	2
13	0.5	1.0	0.8	1.0	1.0	1.0	10	1	67	0.5	1.0	0.8	1.0	0.1	0.1	10	1
14	0.5	1.0	0.8	1.0	1.0	1.0	10	2	68	0.5	1.0	0.8	1.0	0.1	0.1	10	2
15	0.5	1.0	0.8	1.0	1.0	1.0	10	5	69	0.5	1.0	0.8	1.0	0.1	0.1	10	5
16	0.5	1.0	0.8	1.0	1.0	1.0	15	1	70	0.5	1.0	0.8	1.0	0.1	0.1	15	1
17	0.5	1.0	0.8	1.0	1.0	1.0	15	2	71	0.5	1.0	0.8	1.0	0.1	0.1	15	2
18	0.5	1.0	0.8	1.0	1.0	1.0	15	5	72	0.5	1.0	0.8	1.0	0.1	0.1	15	5
19	0.2	1.0	0.8	1.0	1.0	1.0	2	1	73	0.2	1.0	0.8	1.0	0.1	0.1	2	1
20	0.2	1.0	0.8	1.0	1.0	1.0	5	1	74	0.2	1.0	0.8	1.0	0.1	0.1	5	1
21	0.2	1.0	0.8	1.0	1.0	1.0	5	2	75	0.2	1.0	0.8	1.0	0.1	0.1	5	2
22	0.2	1.0	0.8	1.0	1.0	1.0	10	1	76	0.2	1.0	0.8	1.0	0.1	0.1	10	1
23	0.2	1.0	0.8	1.0	1.0	1.0	10	2	77	0.2	1.0	0.8	1.0	0.1	0.1	10	2
24	0.2	1.0	0.8	1.0	1.0	1.0	10	5	78	0.2	1.0	0.8	1.0	0.1	0.1	10	5
25	0.2	1.0	0.8	1.0	1.0	1.0	15	1	79	0.2	1.0	0.8	1.0	0.1	0.1	15	1
26	0.2	1.0	0.8	1.0	1.0	1.0	15	2	80	0.2	1.0	0.8	1.0	0.1	0.1	15	2
27	0.2	1.0	0.8	1.0	1.0	1.0	15	5	81	0.2	1.0	0.8	1.0	0.1	0.1	15	5
28	0.5	1.0	0.5	1.0	1.0	1.0	2	1	82	0.5	1.0	0.5	1.0	0.1	0.1	2	1
29	0.5	1.0	0.5	1.0	1.0	1.0	5	1	83	0.5	1.0	0.5	1.0	0.1	0.1	5	1
30	0.5	1.0	0.5	1.0	1.0	1.0	5	2	84	0.5	1.0	0.5	1.0	0.1	0.1	5	2
31	0.5	1.0	0.5	1.0	1.0	1.0	10	1	85	0.5	1.0	0.5	1.0	0.1	0.1	10	1
32	0.5	1.0	0.5	1.0	1.0	1.0	10	2	86	0.5	1.0	0.5	1.0	0.1	0.1	10	2
33	0.5	1.0	0.5	1.0	1.0	1.0	10	5	87	0.5	1.0	0.5	1.0	0.1	0.1	10	5
34	0.5	1.0	0.5	1.0	1.0	1.0	15	1	88	0.5	1.0	0.5	1.0	0.1	0.1	15	1
35	0.5	1.0	0.5	1.0	1.0	1.0	15	2	89	0.5	1.0	0.5	1.0	0.1	0.1	15	2
36	0.5	1.0	0.5	1.0	1.0	1.0	15	5	90	0.5	1.0	0.5	1.0	0.1	0.1	15	5
37	0.2	1.0	0.5	1.0	1.0	1.0	2	1	91	0.2	1.0	0.5	1.0	0.1	0.1	2	1
38	0.2	1.0	0.5	1.0	1.0	1.0	5	1	92	0.2	1.0	0.5	1.0	0.1	0.1	5	1
39	0.2	1.0	0.5	1.0	1.0	1.0	5	2	93	0.2	1.0	0.5	1.0	0.1	0.1	5	2
40	0.2	1.0	0.5	1.0	1.0	1.0	10	1	94	0.2	1.0	0.5	1.0	0.1	0.1	10	1
41	0.2	1.0	0.5	1.0	1.0	1.0	10	2	95	0.2	1.0	0.5	1.0	0.1	0.1	10	2
42	0.2	1.0	0.5	1.0	1.0	1.0	10	5	96	0.2	1.0	0.5	1.0	0.1	0.1	10	5
43	0.2	1.0	0.5	1.0	1.0	1.0	15	1	97	0.2	1.0	0.5	1.0	0.1	0.1	15	1
44	0.2	1.0	0.5	1.0	1.0	1.0	15	2	98	0.2	1.0	0.5	1.0	0.1	0.1	15	2
45	0.2	1.0	0.5	1.0	1.0	1.0	15	5	99	0.2	1.0	0.5	1.0	0.1	0.1	15	5
46	0.2	1.0	0.2	1.0	1.0	1.0	2	1	100	0.2	1.0	0.2	1.0	0.1	0.1	2	1
47	0.2	1.0	0.2	1.0	1.0	1.0	5	1	101	0.2	1.0	0.2	1.0	0.1	0.1	5	1
48	0.2	1.0	0.2	1.0	1.0	1.0	5	2	102	0.2	1.0	0.2	1.0	0.1	0.1	5	2
49	0.2	1.0	0.2	1.0	1.0	1.0	10	1	103	0.2	1.0	0.2	1.0	0.1	0.1	10	1
50	0.2	1.0	0.2	1.0	1.0	1.0	10	2	104	0.2	1.0	0.2	1.0	0.1	0.1	10	2
51	0.2	1.0	0.2	1.0	1.0	1.0	10	5	105	0.2	1.0	0.2	1.0	0.1	0.1	10	5
52	0.2	1.0	0.2	1.0	1.0	1.0	15	1	106	0.2	1.0	0.2	1.0	0.1	0.1	15	1
53	0.2	1.0	0.2	1.0	1.0	1.0	15	2	107	0.2	1.0	0.2	1.0	0.1	0.1	15	2
54	0.2	1.0	0.2	1.0	1.0	1.0	15	5	108	0.2	1.0	0.2	1.0	0.1	0.1	15	5

Table D.1: G-limited Case: Problem data

NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S	K	NO	λ_1	μ_1	λ_2	μ_2	β_{12}	β_{21}	S	K
109	0.7	1.0	1.4	2.0	1.0	2.0	2	1	163	0.7	1.0	1.4	2.0	0.1	0.2	2	1
110	0.7	1.0	1.4	2.0	1.0	2.0	5	1	164	0.7	1.0	1.4	2.0	0.1	0.2	5	1
111	0.7	1.0	1.4	2.0	1.0	2.0	5	2	165	0.7	1.0	1.4	2.0	0.1	0.2	5	2
112	0.7	1.0	1.4	2.0	1.0	2.0	10	1	166	0.7	1.0	1.4	2.0	0.1	0.2	10	1
113	0.7	1.0	1.4	2.0	1.0	2.0	10	2	167	0.7	1.0	1.4	2.0	0.1	0.2	10	2
114	0.7	1.0	1.4	2.0	1.0	2.0	10	5	168	0.7	1.0	1.4	2.0	0.1	0.2	10	5
115	0.7	1.0	1.4	2.0	1.0	2.0	15	1	169	0.7	1.0	1.4	2.0	0.1	0.2	15	1
116	0.7	1.0	1.4	2.0	1.0	2.0	15	2	170	0.7	1.0	1.4	2.0	0.1	0.2	15	2
117	0.7	1.0	1.4	2.0	1.0	2.0	15	5	171	0.7	1.0	1.4	2.0	0.1	0.2	15	5
118	0.4	1.0	1.4	2.0	1.0	2.0	2	1	172	0.4	1.0	1.4	2.0	0.1	0.2	2	1
119	0.4	1.0	1.4	2.0	1.0	2.0	5	1	173	0.4	1.0	1.4	2.0	0.1	0.2	5	1
120	0.4	1.0	1.4	2.0	1.0	2.0	5	2	174	0.4	1.0	1.4	2.0	0.1	0.2	5	2
121	0.4	1.0	1.4	2.0	1.0	2.0	10	1	175	0.4	1.0	1.4	2.0	0.1	0.2	10	1
122	0.4	1.0	1.4	2.0	1.0	2.0	10	2	176	0.4	1.0	1.4	2.0	0.1	0.2	10	2
123	0.4	1.0	1.4	2.0	1.0	2.0	10	5	177	0.4	1.0	1.4	2.0	0.1	0.2	10	5
124	0.4	1.0	1.4	2.0	1.0	2.0	15	1	178	0.4	1.0	1.4	2.0	0.1	0.2	15	1
125	0.4	1.0	1.4	2.0	1.0	2.0	15	2	179	0.4	1.0	1.4	2.0	0.1	0.2	15	2
126	0.4	1.0	1.4	2.0	1.0	2.0	15	5	180	0.4	1.0	1.4	2.0	0.1	0.2	15	5
127	0.1	1.0	1.4	2.0	1.0	2.0	2	1	181	0.1	1.0	1.4	2.0	0.1	0.2	2	1
128	0.1	1.0	1.4	2.0	1.0	2.0	5	1	182	0.1	1.0	1.4	2.0	0.1	0.2	5	1
129	0.1	1.0	1.4	2.0	1.0	2.0	5	2	183	0.1	1.0	1.4	2.0	0.1	0.2	5	2
130	0.1	1.0	1.4	2.0	1.0	2.0	10	1	184	0.1	1.0	1.4	2.0	0.1	0.2	10	1
131	0.1	1.0	1.4	2.0	1.0	2.0	10	2	185	0.1	1.0	1.4	2.0	0.1	0.2	10	2
132	0.1	1.0	1.4	2.0	1.0	2.0	10	5	186	0.1	1.0	1.4	2.0	0.1	0.2	10	5
133	0.1	1.0	1.4	2.0	1.0	2.0	15	1	187	0.1	1.0	1.4	2.0	0.1	0.2	15	1
134	0.1	1.0	1.4	2.0	1.0	2.0	15	2	188	0.1	1.0	1.4	2.0	0.1	0.2	15	2
135	0.1	1.0	1.4	2.0	1.0	2.0	15	5	189	0.1	1.0	1.4	2.0	0.1	0.2	15	5
136	0.4	1.0	0.8	2.0	1.0	2.0	2	1	190	0.4	1.0	0.8	2.0	0.1	0.2	2	1
137	0.4	1.0	0.8	2.0	1.0	2.0	5	1	191	0.4	1.0	0.8	2.0	0.1	0.2	5	1
138	0.4	1.0	0.8	2.0	1.0	2.0	5	2	192	0.4	1.0	0.8	2.0	0.1	0.2	5	2
139	0.4	1.0	0.8	2.0	1.0	2.0	10	1	193	0.4	1.0	0.8	2.0	0.1	0.2	10	1
140	0.4	1.0	0.8	2.0	1.0	2.0	10	2	194	0.4	1.0	0.8	2.0	0.1	0.2	10	2
141	0.4	1.0	0.8	2.0	1.0	2.0	10	5	195	0.4	1.0	0.8	2.0	0.1	0.2	10	5
142	0.4	1.0	0.8	2.0	1.0	2.0	15	1	196	0.4	1.0	0.8	2.0	0.1	0.2	15	1
143	0.4	1.0	0.8	2.0	1.0	2.0	15	2	197	0.4	1.0	0.8	2.0	0.1	0.2	15	2
144	0.4	1.0	0.8	2.0	1.0	2.0	15	5	198	0.4	1.0	0.8	2.0	0.1	0.2	15	5
145	0.1	1.0	0.8	2.0	1.0	2.0	2	1	199	0.1	1.0	0.8	2.0	0.1	0.2	2	1
146	0.1	1.0	0.8	2.0	1.0	2.0	5	1	200	0.1	1.0	0.8	2.0	0.1	0.2	5	1
147	0.1	1.0	0.8	2.0	1.0	2.0	5	2	201	0.1	1.0	0.8	2.0	0.1	0.2	5	2
148	0.1	1.0	0.8	2.0	1.0	2.0	10	1	202	0.1	1.0	0.8	2.0	0.1	0.2	10	1
149	0.1	1.0	0.8	2.0	1.0	2.0	10	2	203	0.1	1.0	0.8	2.0	0.1	0.2	10	2
150	0.1	1.0	0.8	2.0	1.0	2.0	10	5	204	0.1	1.0	0.8	2.0	0.1	0.2	10	5
151	0.1	1.0	0.8	2.0	1.0	2.0	15	1	205	0.1	1.0	0.8	2.0	0.1	0.2	15	1
152	0.1	1.0	0.8	2.0	1.0	2.0	15	2	206	0.1	1.0	0.8	2.0	0.1	0.2	15	2
153	0.1	1.0	0.8	2.0	1.0	2.0	15	5	207	0.1	1.0	0.8	2.0	0.1	0.2	15	5
154	0.1	1.0	0.2	2.0	1.0	2.0	2	1	208	0.1	1.0	0.2	2.0	0.1	0.2	2	1
155	0.1	1.0	0.2	2.0	1.0	2.0	5	1	209	0.1	1.0	0.2	2.0	0.1	0.2	5	1
156	0.1	1.0	0.2	2.0	1.0	2.0	5	2	210	0.1	1.0	0.2	2.0	0.1	0.2	5	2
157	0.1	1.0	0.2	2.0	1.0	2.0	10	1	211	0.1	1.0	0.2	2.0	0.1	0.2	10	1
158	0.1	1.0	0.2	2.0	1.0	2.0	10	2	212	0.1	1.0	0.2	2.0	0.1	0.2	10	2
159	0.1	1.0	0.2	2.0	1.0	2.0	10	5	213	0.1	1.0	0.2	2.0	0.1	0.2	10	5
160	0.1	1.0	0.2	2.0	1.0	2.0	15	1	214	0.1	1.0	0.2	2.0	0.1	0.2	15	1
161	0.1	1.0	0.2	2.0	1.0	2.0	15	2	215	0.1	1.0	0.2	2.0	0.1	0.2	15	2
162	0.1	1.0	0.2	2.0	1.0	2.0	15	5	216	0.1	1.0	0.2	2.0	0.1	0.2	15	5

Table D.2: G-limited Case: Problem data *(continued)*

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.24837	0.24837	1.65616	1.65616	0.24846	0.24846	1.65619	1.65619	0.04	0.04	0.00	0.00
2	0.24999	0.24999	4.64564	4.64564	0.24999	0.24999	4.64564	4.64564	0.00	0.00	0.00	0.00
3	0.33317	0.33317	4.39175	4.39175	0.33316	0.33316	4.39196	4.39196	0.00	0.00	0.00	0.00
4	0.25000	0.25000	9.64560	9.64560	0.25000	0.25000	9.64560	9.64560	0.00	0.00	0.00	0.00
5	0.33333	0.33333	9.38900	9.38900	0.33333	0.33333	9.38900	9.38900	0.00	0.00	0.00	0.00
6	0.41663	0.41663	8.88624	8.88624	0.41662	0.41662	8.88651	8.88651	0.00	0.00	0.00	0.00
7	0.25000	0.25000	14.6456	14.6456	0.25000	0.25000	14.6456	14.6456	0.00	0.00	0.00	0.00
8	0.33333	0.33333	14.3890	14.3890	0.33333	0.33333	14.3890	14.3890	0.00	0.00	0.00	0.00
9	0.41666	0.41666	13.8846	13.8846	0.41666	0.41666	13.8846	13.8846	0.00	0.00	0.00	0.00
10	0.23882	0.25112	1.41064	1.65060	0.23900	0.25162	1.41028	1.65037	0.07	0.20	0.03	0.01
11	0.24969	0.25009	4.29081	4.64537	0.24969	0.25010	4.29082	4.64541	0.00	0.00	0.00	0.00
12	0.32714	0.33606	3.68223	4.38073	0.32716	0.33585	3.68259	4.38383	0.00	0.06	0.01	0.07
13	0.24999	0.25000	9.28377	9.64560	0.24999	0.25000	9.28377	9.64559	0.00	0.00	0.00	0.00
14	0.33307	0.33346	8.49955	9.38845	0.33307	0.33345	8.49955	9.38862	0.00	0.00	0.00	0.00
15	0.40717	0.42324	6.92626	8.84270	0.40716	0.42191	6.92800	8.86076	0.00	0.32	0.03	0.20
16	0.25000	0.25000	14.2837	14.6456	0.25000	0.25000	14.2837	14.6456	0.00	0.00	0.00	0.00
17	0.33332	0.33333	13.4845	14.3889	0.33332	0.33335	13.4845	14.3889	0.00	0.01	0.00	0.00
18	0.41427	0.41837	11.3323	13.8724	0.41427	0.41790	11.3324	13.8784	0.00	0.11	0.00	0.04
19	0.16687	0.27252	0.67949	1.60939	0.16704	0.27580	0.67738	1.60556	0.10	1.19	0.31	0.24
20	0.19187	0.26934	1.49163	4.59914	0.19187	0.27240	1.49160	4.59721	0.00	1.12	0.00	0.04
21	0.19736	0.39842	0.97442	4.15425	0.19738	0.40584	0.97270	4.14980	0.01	1.83	0.18	0.11
22	0.19875	0.26708	2.13554	9.60435	0.19875	0.27059	2.13554	9.60109	0.00	1.30	0.00	0.03
23	0.19993	0.39999	1.07639	9.11726	0.19993	0.40716	1.07638	9.12197	0.00	1.76	0.00	0.05
24	0.19997	0.56399	1.22143	7.78277	0.19997	0.57230	1.22435	7.81617	0.00	1.45	0.24	0.43
25	0.19978	0.26673	2.33351	14.6051	0.19978	0.27021	2.33351	14.6019	0.00	1.29	0.00	0.02
26	0.19999	0.40000	1.08127	14.1162	0.19999	0.40716	1.08127	14.1215	0.00	1.76	0.00	0.04
27	0.20000	0.57006	1.23441	12.5370	0.20000	0.57679	1.23558	12.6215	0.00	1.17	0.09	0.67
28	0.24090	0.24090	1.40233	1.40233	0.24171	0.24171	1.40046	1.40046	0.34	0.34	0.13	0.13
29	0.24979	0.24979	4.29006	4.29006	0.24979	0.24979	4.29024	4.29024	0.00	0.00	0.00	0.00
30	0.32935	0.32935	3.65588	3.65588	0.32959	0.32959	3.66020	3.66020	0.07	0.07	0.12	0.12
31	0.24999	0.24999	9.28376	9.28376	0.24999	0.24999	9.28377	9.28377	0.00	0.00	0.00	0.00
32	0.33319	0.33319	8.49693	8.49693	0.33318	0.33318	8.49809	8.49809	0.00	0.00	0.01	0.01
33	0.41165	0.41165	6.75451	6.75451	0.41172	0.41172	6.80815	6.80815	0.02	0.02	0.79	0.79
34	0.25000	0.25000	14.2837	14.2837	0.25000	0.25000	14.2837	14.2837	0.00	0.00	0.00	0.00
35	0.33332	0.33332	13.4843	13.4843	0.33332	0.33332	13.4844	13.4844	0.00	0.00	0.00	0.00
36	0.41562	0.41562	11.2439	11.2439	0.41542	0.41542	11.2857	11.2857	0.05	0.05	0.37	0.37
37	0.16734	0.25739	0.67180	1.33967	0.16798	0.26138	0.66370	1.32864	0.38	1.53	1.22	0.83
38	0.19188	0.26833	1.49009	4.16170	0.19191	0.27168	1.48852	4.15794	0.01	1.23	0.11	0.09
39	0.19746	0.37508	0.95135	3.13941	0.19769	0.38522	0.92609	3.07264	0.12	2.63	2.73	2.17
40	0.19875	0.26707	2.13551	9.15203	0.19875	0.27062	2.13549	9.15044	0.00	1.31	0.00	0.02
41	0.19993	0.39416	1.06993	7.22941	0.19993	0.40277	1.06427	7.18121	0.00	2.14	0.53	0.67
42	0.19997	0.47670	1.05873	3.98627	0.19999	0.48790	0.92186	3.43355	0.01	2.30	14.8	16.1
43	0.19978	0.26673	2.33351	14.1542	0.19978	0.27021	2.33351	14.1531	0.00	1.29	0.00	0.01
44	0.19999	0.39847	1.07956	11.8208	0.19999	0.40615	1.07854	11.8183	0.00	1.89	0.09	0.02
45	0.20000	0.48941	1.08349	5.17858	0.20000	0.49668	0.93254	4.03957	0.00	1.46	16.1	28.2
46	0.17097	0.17097	0.61600	0.61600	0.17292	0.17292	0.59092	0.59092	1.13	1.13	4.24	4.24
47	0.19371	0.19371	1.29891	1.29891	0.19513	0.19513	1.18483	1.18483	0.73	0.73	9.63	9.63
48	0.19856	0.19856	0.72755	0.72755	0.19914	0.19914	0.63724	0.63724	0.29	0.29	14.1	14.1
49	0.19922	0.19922	1.76187	1.76187	0.19961	0.19961	1.49498	1.49498	0.19	0.19	17.8	17.8
50	0.19997	0.19997	0.77168	0.77168	0.19999	0.19999	0.65698	0.65698	0.01	0.01	17.4	17.4
51	0.19999	0.19999	0.63604	0.63604	0.19999	0.19999	0.56176	0.56176	0.00	0.00	13.2	13.2
52	0.19989	0.19989	1.87271	1.87271	0.19996	0.19996	1.54446	1.54446	0.04	0.04	21.2	21.2
53	0.20000	0.20000	0.77296	0.77296	0.20000	0.20000	0.65721	0.65721	0.00	0.00	17.6	17.6
54	0.20000	0.20000	0.63614	0.63614	0.20000	0.20000	0.56177	0.56177	0.00	0.00	13.2	13.2

Table D.3: G-limited Case: Exact and decomposed

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
55	0.04545	0.04545	1.94296	1.94296	0.04545	0.04545	1.94296	1.94296	0.00	0.00	0.00	0.00
56	0.04545	0.04545	4.94296	4.94296	0.04545	0.04545	4.94296	4.94296	0.00	0.00	0.00	0.00
57	0.08333	0.08333	4.86622	4.86622	0.08333	0.08333	4.86622	4.86622	0.00	0.00	0.00	0.00
58	0.04545	0.04545	9.94296	9.94296	0.04545	0.04545	9.94296	9.94296	0.00	0.00	0.00	0.00
59	0.08333	0.08333	9.86622	9.86622	0.08333	0.08333	9.86622	9.86622	0.00	0.00	0.00	0.00
60	0.16666	0.16666	9.59375	9.59375	0.16666	0.16666	9.59375	9.59375	0.00	0.00	0.00	0.00
61	0.04545	0.04545	14.9429	14.9429	0.04545	0.04545	14.9429	14.9429	0.00	0.00	0.00	0.00
62	0.08333	0.08333	14.8662	14.8662	0.08333	0.08333	14.8662	14.8662	0.00	0.00	0.00	0.00
63	0.16666	0.16666	14.5937	14.5937	0.16666	0.16666	14.5937	14.5937	0.00	0.00	0.00	0.00
64	0.04544	0.04545	1.90796	1.94296	0.04544	0.04545	1.90796	1.94296	0.00	0.00	0.00	0.00
65	0.04545	0.04545	4.90792	4.94296	0.04545	0.04545	4.90792	4.94296	0.00	0.00	0.00	0.00
66	0.08333	0.08333	4.77300	4.86622	0.08333	0.08333	4.77300	4.86622	0.00	0.00	0.00	0.00
67	0.04545	0.04545	9.90792	9.94296	0.04545	0.04545	9.90792	9.94296	0.00	0.00	0.00	0.00
68	0.08333	0.08333	9.77298	9.86622	0.08333	0.08333	9.77298	9.86622	0.00	0.00	0.00	0.00
69	0.16665	0.16666	9.23084	9.59374	0.16665	0.16666	9.23084	9.59375	0.00	0.00	0.00	0.00
70	0.04545	0.04545	14.9079	14.9429	0.04545	0.04545	14.9079	14.9429	0.00	0.00	0.00	0.00
71	0.08333	0.08333	14.7729	14.8662	0.08333	0.08333	14.7729	14.8662	0.00	0.00	0.00	0.00
72	0.16666	0.16666	14.2305	14.5937	0.16666	0.16666	14.2305	14.5937	0.00	0.00	0.00	0.00
73	0.04515	0.04546	1.75564	1.94294	0.04515	0.04546	1.75564	1.94294	0.00	0.00	0.00	0.00
74	0.04545	0.04545	4.75025	4.94296	0.04545	0.04545	4.75025	4.94296	0.00	0.00	0.00	0.00
75	0.08298	0.08336	4.28452	4.86616	0.08298	0.08336	4.28452	4.86617	0.00	0.00	0.00	0.00
76	0.04545	0.04545	9.75024	9.94296	0.04545	0.04545	9.75024	9.94296	0.00	0.00	0.00	0.00
77	0.08333	0.08333	9.26866	9.86622	0.08333	0.08333	9.26866	9.86622	0.00	0.00	0.00	0.00
78	0.15809	0.16838	6.43116	9.58859	0.15809	0.16828	6.43116	9.58935	0.00	0.06	0.00	0.01
79	0.04545	0.04545	14.7502	14.9429	0.04545	0.04545	14.7502	14.9429	0.00	0.00	0.00	0.00
80	0.08333	0.08333	14.2685	14.8662	0.08333	0.08333	14.2685	14.8662	0.00	0.00	0.00	0.00
81	0.16351	0.16729	10.4073	14.5918	0.16351	0.16726	10.4073	14.5921	0.00	0.02	0.00	0.00
82	0.04544	0.04544	1.90796	1.90796	0.04544	0.04544	1.90796	1.90796	0.00	0.00	0.00	0.00
83	0.04545	0.04545	4.90792	4.90792	0.04545	0.04545	4.90792	4.90792	0.00	0.00	0.00	0.00
84	0.08333	0.08333	4.77300	4.77300	0.08333	0.08333	4.77300	4.77300	0.00	0.00	0.00	0.00
85	0.04545	0.04545	9.90792	9.90792	0.04545	0.04545	9.90792	9.90792	0.00	0.00	0.00	0.00
86	0.08333	0.08333	9.77298	9.77298	0.08333	0.08333	9.77298	9.77298	0.00	0.00	0.00	0.00
87	0.16665	0.16665	9.23082	9.23082	0.16665	0.16665	9.23083	9.23083	0.00	0.00	0.00	0.00
88	0.04545	0.04545	14.9079	14.9079	0.04545	0.04545	14.9079	14.9079	0.00	0.00	0.00	0.00
89	0.08333	0.08333	14.7729	14.7729	0.08333	0.08333	14.7729	14.7729	0.00	0.00	0.00	0.00
90	0.16666	0.16666	14.2305	14.2305	0.16666	0.16666	14.2305	14.2305	0.00	0.00	0.00	0.00
91	0.04515	0.04546	1.75563	1.90792	0.04515	0.04546	1.75563	1.90793	0.00	0.00	0.00	0.00
92	0.04545	0.04545	4.75025	4.90792	0.04545	0.04545	4.75025	4.90791	0.00	0.00	0.00	0.00
93	0.08298	0.08336	4.28451	4.77287	0.08298	0.08336	4.28452	4.77290	0.00	0.00	0.00	0.00
94	0.04545	0.04545	9.75024	9.90792	0.04545	0.04545	9.75024	9.90792	0.00	0.00	0.00	0.00
95	0.08333	0.08333	9.26866	9.77298	0.08333	0.08333	9.26866	9.77298	0.00	0.00	0.00	0.00
96	0.15809	0.16835	6.43111	9.21668	0.15809	0.16827	6.43106	9.22026	0.00	0.05	0.00	0.04
97	0.04545	0.04545	14.7502	14.9079	0.04545	0.04545	14.7502	14.9079	0.00	0.00	0.00	0.00
98	0.08333	0.08333	14.2685	14.7729	0.08333	0.08333	14.2685	14.7729	0.00	0.00	0.00	0.00
99	0.16351	0.16729	10.4073	14.2251	0.16351	0.16724	10.4073	14.2267	0.00	0.03	0.00	0.01
100	0.04516	0.04516	1.75554	1.75554	0.04516	0.04516	1.75554	1.75554	0.01	0.01	0.00	0.00
101	0.04545	0.04545	4.75025	4.75025	0.04545	0.04545	4.75025	4.75025	0.00	0.00	0.00	0.00
102	0.08301	0.08301	4.28394	4.28394	0.08301	0.08301	4.28406	4.28406	0.01	0.01	0.00	0.00
103	0.04545	0.04545	9.75024	9.75024	0.04545	0.04545	9.75024	9.75024	0.00	0.00	0.00	0.00
104	0.08333	0.08333	9.26865	9.26865	0.08333	0.08333	9.26865	9.26865	0.00	0.00	0.00	0.00
105	0.15863	0.15863	6.37748	6.37748	0.15909	0.15909	6.36327	6.36327	0.29	0.29	0.22	0.22
106	0.04545	0.04545	14.7502	14.7502	0.04545	0.04545	14.7502	14.7502	0.00	0.00	0.00	0.00
107	0.08333	0.08333	14.2685	14.2685	0.08333	0.08333	14.2685	14.2685	0.00	0.00	0.00	0.00
108	0.16379	0.16379	10.3589	10.3589	0.16396	0.16396	10.3570	10.3570	0.10	0.10	0.02	0.02

Table D.4: G-limited Case: Exact and decomposed *(continued)*

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
109	0.32012	0.33832	1.44041	1.74106	0.32021	0.33914	1.44029	1.74066	0.03	0.24	0.01	0.02
110	0.33304	0.33348	4.34012	4.74405	0.33304	0.33349	4.34012	4.74406	0.00	0.00	0.00	0.00
111	0.43804	0.44942	3.79999	4.55386	0.43804	0.44876	3.80004	4.55609	0.00	0.15	0.00	0.05
112	0.33333	0.33333	9.33540	9.74423	0.33333	0.33335	9.33540	9.74421	0.00	0.00	0.00	0.00
113	0.44424	0.44460	8.67128	9.56160	0.44424	0.44459	8.67128	9.56166	0.00	0.00	0.00	0.00
114	0.54701	0.56614	7.37204	9.14736	0.54701	0.56275	7.37230	9.15946	0.00	0.60	0.00	0.13
115	0.33333	0.33333	14.3353	14.7442	0.33333	0.33334	14.3353	14.7442	0.00	0.00	0.00	0.00
116	0.44443	0.44444	13.6635	14.5618	0.44443	0.44447	13.6635	14.5618	0.00	0.01	0.00	0.00
117	0.55391	0.55760	12.0079	14.1689	0.55391	0.55681	12.0079	14.1717	0.00	0.14	0.00	0.02
118	0.27713	0.35844	1.00677	1.72096	0.27727	0.36354	1.00605	1.71707	0.05	1.40	0.07	0.23
119	0.31758	0.34120	3.05445	4.73522	0.31758	0.34241	3.05445	4.73485	0.00	0.35	0.00	0.01
120	0.36590	0.50615	2.02012	4.45976	0.36593	0.51910	2.01949	4.45046	0.01	2.50	0.03	0.21
121	0.33021	0.33489	7.19922	9.74244	0.33021	0.33515	7.19922	9.74236	0.00	0.08	0.00	0.00
122	0.38871	0.48902	3.54849	9.48385	0.38871	0.50268	3.54850	9.47385	0.00	2.72	0.00	0.11
123	0.39740	0.74949	2.49639	8.59921	0.39741	0.77310	2.49528	8.59964	0.00	3.05	0.04	0.01
124	0.33263	0.33368	11.8628	14.7438	0.33263	0.33396	11.8628	14.7435	0.00	0.08	0.00	0.00
125	0.39536	0.48371	4.59071	14.4931	0.39536	0.49909	4.59071	14.4794	0.00	3.08	0.00	0.09
126	0.39971	0.74995	2.63231	13.5505	0.39971	0.77346	2.63249	13.5767	0.00	3.04	0.01	0.19
127	0.09775	0.44278	0.19543	1.63865	0.09776	0.44677	0.19477	1.63591	0.01	0.89	0.34	0.17
128	0.09998	0.44996	0.21933	4.61407	0.09998	0.45365	0.21933	4.61309	0.00	0.81	0.00	0.02
129	0.09999	0.71309	0.20528	4.09913	0.09999	0.71972	0.20490	4.09756	0.00	0.92	0.19	0.04
130	0.10000	0.45000	0.21964	9.61379	0.10000	0.45368	0.21964	9.61289	0.00	0.81	0.00	0.01
131	0.10000	0.71989	0.20556	9.04334	0.10000	0.72569	0.20555	9.04832	0.00	0.80	0.00	0.06
132	0.10000	1.08945	0.28235	7.06257	0.10000	1.10071	0.28150	7.05812	0.00	1.02	0.30	0.06
133	0.10000	0.45000	0.21964	14.6137	0.10000	0.45368	0.21964	14.6128	0.00	0.81	0.00	0.01
134	0.10000	0.71999	0.20556	14.0416	0.10000	0.72577	0.20556	14.0470	0.00	0.80	0.00	0.04
135	0.10000	1.11406	0.28546	11.3840	0.10000	1.12351	0.28553	11.4091	0.00	0.84	0.02	0.22
136	0.27765	0.34448	1.00303	1.47649	0.27839	0.35043	0.99945	1.46590	0.27	1.70	0.36	0.72
137	0.31759	0.34096	3.05416	4.43934	0.31760	0.34225	3.05411	4.44055	0.00	0.38	0.00	0.03
138	0.36612	0.49175	2.00970	3.72582	0.36669	0.50873	1.99754	3.67542	0.16	3.34	0.61	1.37
139	0.33021	0.33489	7.19922	9.45969	0.33021	0.33506	7.19922	9.46046	0.00	0.05	0.00	0.01
140	0.38871	0.48787	3.54705	8.59397	0.38872	0.50243	3.54624	8.59206	0.00	2.90	0.02	0.02
141	0.39754	0.68177	2.39084	5.72408	0.39813	0.72236	2.26725	5.21302	0.15	5.62	5.45	9.80
142	0.33263	0.33368	11.8628	14.4642	0.33263	0.33382	11.8628	14.4641	0.00	0.04	0.00	0.00
143	0.39536	0.48360	4.59052	13.6034	0.39536	0.49908	4.59056	13.6077	0.00	3.10	0.00	0.03
144	0.39970	0.70450	2.55790	8.79050	0.39978	0.74351	2.47608	7.98172	0.02	5.25	3.30	10.1
145	0.09776	0.41179	0.19450	1.33186	0.09781	0.41663	0.19176	1.32368	0.06	1.16	1.43	0.62
146	0.09998	0.44704	0.21925	4.06215	0.09998	0.45114	0.21901	4.06228	0.00	0.91	0.11	0.00
147	0.09999	0.64238	0.20215	2.78592	0.09999	0.65145	0.19722	2.74707	0.00	1.39	2.50	1.41
148	0.10000	0.44995	0.21964	9.01345	0.10000	0.45365	0.21963	9.01998	0.00	0.82	0.00	0.07
149	0.10000	0.69098	0.20432	6.08999	0.10000	0.69869	0.20273	6.03055	0.00	1.10	0.78	0.99
150	0.10000	0.78824	0.24227	2.71653	0.10000	0.79264	0.21371	2.51252	0.00	0.55	13.3	8.12
151	0.10000	0.44999	0.21964	14.0120	0.10000	0.45368	0.21964	14.0189	0.00	0.81	0.00	0.05
152	0.10000	0.70653	0.20499	9.83031	0.10000	0.71333	0.20434	9.75705	0.00	0.95	0.32	0.75
153	0.10000	0.79765	0.24352	2.98670	0.10000	0.79904	0.21413	2.67083	0.00	0.17	13.7	11.8
154	0.09799	0.18642	0.17916	0.37507	0.09813	0.18725	0.17226	0.36220	0.15	0.44	4.01	3.55
155	0.09999	0.19944	0.19972	0.51357	0.09999	0.19958	0.19018	0.48208	0.00	0.07	5.02	6.53
156	0.09999	0.19986	0.16813	0.34682	0.09999	0.19989	0.15878	0.33130	0.00	0.02	5.89	4.68
157	0.10000	0.19999	0.19999	0.52652	0.10000	0.19999	0.19037	0.49089	0.00	0.00	5.06	7.26
158	0.10000	0.20000	0.16819	0.34838	0.10000	0.20000	0.15882	0.33231	0.00	0.00	5.90	4.84
159	0.10000	0.20000	0.16602	0.32229	0.10000	0.20000	0.15627	0.31140	0.00	0.00	6.24	3.50
160	0.10000	0.20000	0.19999	0.52666	0.10000	0.20000	0.19037	0.49095	0.00	0.00	5.06	7.27
161	0.10000	0.20000	0.16819	0.34838	0.10000	0.20000	0.15882	0.33231	0.00	0.00	5.90	4.84
162	0.10000	0.20000	0.16602	0.32229	0.10000	0.20000	0.15627	0.31140	0.00	0.00	6.24	3.50

Table D.5: G-limited Case: Exact and decomposed *(continued)*

	EXACT				DECOMPOSITION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
163	0.06059	0.06060	1.91236	1.95662	0.06059	0.06060	1.91236	1.95662	0.00	0.00	0.00	0.00
164	0.06060	0.06060	4.91232	4.95662	0.06060	0.06060	4.91232	4.95662	0.00	0.00	0.00	0.00
165	0.11110	0.11111	4.79025	4.89706	0.11428	0.11764	4.78293	4.89073	2.78	5.56	0.15	0.13
166	0.06060	0.06060	9.91232	9.95662	0.06060	0.06060	9.91232	9.95662	0.00	0.00	0.00	0.00
167	0.11111	0.11111	9.79024	9.89706	0.11111	0.11111	9.79024	9.89706	0.00	0.00	0.00	0.00
168	0.22221	0.22222	9.33365	9.67863	0.22221	0.22222	9.33365	9.67863	0.00	0.00	0.00	0.00
169	0.06060	0.06060	14.9123	14.9566	0.06060	0.06060	14.9123	14.9566	0.00	0.00	0.00	0.00
170	0.11111	0.11111	14.7902	14.8970	0.11111	0.11111	14.7902	14.8970	0.00	0.00	0.00	0.00
171	0.22222	0.22222	14.3334	14.6786	0.22222	0.22222	14.3334	14.6786	0.00	0.00	0.00	0.00
172	0.06050	0.06061	1.84247	1.95661	0.06050	0.06061	1.84247	1.95661	0.00	0.00	0.00	0.00
173	0.06060	0.06060	4.84165	4.95662	0.06060	0.06060	4.84165	4.95662	0.00	0.00	0.00	0.00
174	0.11105	0.11111	4.58874	4.89706	0.11105	0.11111	4.58874	4.89706	0.00	0.00	0.00	0.00
175	0.06060	0.06060	9.84165	9.95662	0.06060	0.06060	9.84165	9.95662	0.00	0.00	0.00	0.00
176	0.11111	0.11111	9.58770	9.89706	0.11111	0.11111	9.58770	9.89706	0.00	0.00	0.00	0.00
177	0.22120	0.22251	8.36757	9.67817	0.22120	0.22248	8.36757	9.67824	0.00	0.01	0.00	0.00
178	0.06060	0.06060	14.8416	14.9566	0.06060	0.06060	14.8416	14.9566	0.00	0.00	0.00	0.00
179	0.11111	0.11111	14.5877	14.8970	0.11111	0.11111	14.5877	14.8970	0.00	0.00	0.00	0.00
180	0.22216	0.22223	13.3208	14.6786	0.22216	0.22225	13.3208	14.6785	0.00	0.01	0.00	0.00
181	0.05369	0.06105	1.28897	1.95628	0.05369	0.06106	1.28897	1.95627	0.00	0.02	0.00	0.00
182	0.05990	0.06065	3.91530	4.95658	0.05990	0.06067	3.91530	4.95656	0.00	0.04	0.00	0.00
183	0.08778	0.11402	2.17583	4.89413	0.08778	0.11409	2.17583	4.89412	0.00	0.06	0.00	0.00
184	0.06058	0.06060	8.81914	9.95662	0.06058	0.06064	8.81914	9.95659	0.00	0.07	0.00	0.00
185	0.09542	0.11307	3.91059	9.89509	0.09542	0.11305	3.91059	9.89515	0.00	0.02	0.00	0.00
186	0.09977	0.25720	1.55423	9.62260	0.09977	0.25780	1.55423	9.62270	0.00	0.23	0.00	0.00
187	0.06060	0.06060	13.8141	14.9566	0.06060	0.06063	13.8141	14.9565	0.00	0.05	0.00	0.00
188	0.09784	0.11277	5.23252	14.8953	0.09784	0.11226	5.23252	14.8959	0.00	0.45	0.00	0.00
189	0.09998	0.25714	1.57184	14.6226	0.09998	0.25775	1.57184	14.6227	0.00	0.23	0.00	0.00
190	0.06050	0.06060	1.84247	1.92356	0.06050	0.06060	1.84247	1.92356	0.00	0.00	0.00	0.00
191	0.06060	0.06060	4.84165	4.92355	0.06060	0.06060	4.84165	4.92355	0.00	0.00	0.00	0.00
192	0.11105	0.11111	4.58874	4.80892	0.11105	0.11111	4.58874	4.80892	0.00	0.00	0.00	0.00
193	0.06060	0.06060	9.84165	9.92355	0.06060	0.06060	9.84165	9.92355	0.00	0.00	0.00	0.00
194	0.11111	0.11111	9.58770	9.80893	0.11111	0.11111	9.58770	9.80893	0.00	0.00	0.00	0.00
195	0.22120	0.22250	8.36756	9.33560	0.22120	0.22247	8.36756	9.33597	0.00	0.01	0.00	0.00
196	0.06060	0.06060	14.8416	14.9235	0.06060	0.06060	14.8416	14.9235	0.00	0.00	0.00	0.00
197	0.11111	0.11111	14.5877	14.8089	0.11111	0.11111	14.5877	14.8089	0.00	0.00	0.00	0.00
198	0.22216	0.22223	13.3208	14.3368	0.22216	0.22223	13.3208	14.3368	0.00	0.00	0.00	0.00
199	0.05369	0.06104	1.28897	1.92292	0.05369	0.06105	1.28897	1.92293	0.00	0.02	0.00	0.00
200	0.05990	0.06065	3.91530	4.92348	0.05990	0.06066	3.91530	4.92347	0.00	0.02	0.00	0.00
201	0.08778	0.11402	2.17583	4.80263	0.08778	0.11409	2.17583	4.80281	0.00	0.06	0.00	0.00
202	0.06058	0.06060	8.81914	9.92355	0.06058	0.06062	8.81914	9.92352	0.00	0.03	0.00	0.00
203	0.09542	0.11307	3.91059	9.80467	0.09542	0.11310	3.91059	9.80483	0.00	0.03	0.00	0.00
204	0.09977	0.25702	1.55427	9.18454	0.09977	0.25770	1.55417	9.18929	0.00	0.26	0.01	0.05
205	0.06060	0.06060	13.8141	14.9235	0.06060	0.06062	13.8141	14.9235	0.00	0.03	0.00	0.00
206	0.09784	0.11277	5.23252	14.8053	0.09784	0.11277	5.23252	14.8055	0.00	0.00	0.00	0.00
207	0.09998	0.25714	1.57184	14.1817	0.09998	0.25775	1.57184	14.1878	0.00	0.24	0.00	0.04
208	0.05369	0.05978	1.28878	1.65899	0.05370	0.05983	1.28856	1.65877	0.03	0.09	0.02	0.01
209	0.05990	0.06064	3.91530	4.63879	0.05990	0.06064	3.91530	4.63887	0.00	0.00	0.00	0.00
210	0.08778	0.11101	2.17519	3.83063	0.08782	0.11138	2.17264	3.82884	0.04	0.33	0.12	0.05
211	0.06058	0.06060	8.81914	9.63913	0.06058	0.06061	8.81914	9.63910	0.00	0.01	0.00	0.00
212	0.09542	0.11294	3.91054	8.68546	0.09542	0.11303	3.91019	8.69773	0.00	0.08	0.01	0.14
213	0.09976	0.18758	1.54852	3.82025	0.09978	0.18881	1.51543	3.69885	0.02	0.66	2.18	3.28
214	0.06060	0.06060	13.8141	14.6391	0.06060	0.06060	13.8141	14.6391	0.00	0.00	0.00	0.00
215	0.09784	0.11276	5.23251	13.6826	0.09784	0.11282	5.23249	13.6967	0.00	0.05	0.00	0.10
216	0.09998	0.19505	1.56719	4.80881	0.09999	0.19595	1.53524	4.57640	0.00	0.46	2.08	5.08

Table D.6: G-limited Case: Exact and decomposed *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
1	0.24837	0.24837	1.65616	1.65616	0.24842	0.24842	1.65478	1.65478	0.02	0.02	0.08	0.08
2	0.24999	0.24999	4.64564	4.64564	0.24999	0.24999	4.64562	4.64562	0.00	0.00	0.00	0.00
3	0.33317	0.33317	4.39175	4.39175	0.33320	0.33320	4.46439	4.46439	0.01	0.01	1.63	1.63
4	0.25000	0.25000	9.64560	9.64560	0.25000	0.25000	9.64559	9.64559	0.00	0.00	0.00	0.00
5	0.33333	0.33333	9.38900	9.38900	0.33333	0.33333	9.46149	9.46149	0.00	0.00	0.77	0.77
6	0.41663	0.41663	8.88624	8.88624	0.41664	0.41664	9.16191	9.16191	0.00	0.00	3.01	3.01
7	0.25000	0.25000	14.6456	14.6456	0.25000	0.25000	14.6455	14.6455	0.00	0.00	0.00	0.00
8	0.33333	0.33333	14.3890	14.3890	0.33333	0.33333	14.4614	14.4614	0.00	0.00	0.50	0.50
9	0.41666	0.41666	13.8846	13.8846	0.41666	0.41666	14.1598	14.1598	0.00	0.00	1.94	1.94
10	0.23882	0.25112	1.41064	1.65060	0.23827	0.25156	1.39825	1.64875	0.23	0.18	0.89	0.11
11	0.24969	0.25009	4.29081	4.64537	0.24969	0.25009	4.28911	4.64541	0.00	0.00	0.04	0.00
12	0.32714	0.33606	3.68223	4.38073	0.32776	0.33589	3.71793	4.45682	0.19	0.05	0.96	1.71
13	0.24999	0.25000	9.28377	9.64560	0.24999	0.25000	9.27742	9.64558	0.00	0.00	0.07	0.00
14	0.33307	0.33346	8.49955	9.38845	0.33311	0.33344	8.54844	9.46117	0.01	0.01	0.57	0.77
15	0.40717	0.42324	6.92626	8.84270	0.40920	0.42193	7.06424	9.13956	0.50	0.31	1.95	3.25
16	0.25000	0.25000	14.2837	14.6456	0.24999	0.25000	14.2551	14.6455	0.00	0.00	0.20	0.00
17	0.33332	0.33333	13.4845	14.3889	0.33332	0.33334	13.5267	14.4614	0.00	0.00	0.31	0.50
18	0.41427	0.41837	11.3323	13.8724	0.41494	0.41789	11.4854	14.1545	0.16	0.11	1.33	1.99
19	0.16687	0.27252	0.67949	1.60939	0.15932	0.27562	0.62450	1.60184	4.74	1.13	8.81	0.47
20	0.19187	0.26934	1.49163	4.59914	0.18267	0.27242	1.24391	4.59710	5.03	1.13	19.9	0.04
21	0.19736	0.39842	0.97442	4.15425	0.18423	0.40621	0.84216	4.23457	7.13	1.92	15.7	1.90
22	0.19875	0.26708	2.13554	9.60435	0.18804	0.27065	1.61547	9.60098	5.69	1.32	32.1	0.04
23	0.19993	0.39999	1.07639	9.11726	0.18564	0.40716	0.88216	9.21035	7.70	1.76	22.0	1.01
24	0.19997	0.56399	1.22143	7.78277	0.19121	0.57475	1.17510	8.17950	4.58	1.87	3.94	4.85
25	0.19978	0.26673	2.33351	14.6051	0.18935	0.27021	1.73286	14.6019	5.51	1.29	34.6	0.02
26	0.19999	0.40000	1.08127	14.1162	0.18565	0.40717	0.88283	14.2099	7.72	1.76	22.4	0.66
27	0.20000	0.57006	1.23441	12.5370	0.19141	0.57719	1.18532	12.9991	4.49	1.23	4.14	3.55
28	0.24090	0.24090	1.40233	1.40233	0.24087	0.24087	1.38730	1.38730	0.01	0.01	1.08	1.08
29	0.24979	0.24979	4.29006	4.29006	0.24979	0.24979	4.28868	4.28868	0.00	0.00	0.03	0.03
30	0.32935	0.32935	3.65588	3.65588	0.33023	0.33023	3.69412	3.69412	0.27	0.27	1.04	1.04
31	0.24999	0.24999	9.28376	9.28376	0.24999	0.24999	9.28373	9.28373	0.00	0.00	0.00	0.00
32	0.33319	0.33319	8.49693	8.49693	0.33322	0.33322	8.54806	8.54806	0.01	0.01	0.60	0.60
33	0.41165	0.41165	6.75451	6.75451	0.41410	0.41410	6.93712	6.93712	0.59	0.59	2.63	2.63
34	0.25000	0.25000	14.2837	14.2837	0.25000	0.25000	14.2836	14.2836	0.00	0.00	0.00	0.00
35	0.33332	0.33332	13.4843	13.4843	0.33332	0.33332	13.5355	13.5355	0.00	0.00	0.38	0.38
36	0.41562	0.41562	11.2439	11.2439	0.41614	0.41614	11.4386	11.4386	0.12	0.12	1.70	1.70
37	0.16734	0.25739	0.67180	1.33967	0.15982	0.25947	0.60999	1.30558	4.71	0.80	10.1	2.61
38	0.19188	0.26833	1.49009	4.16170	0.18267	0.27168	1.24062	4.15317	5.04	1.23	20.1	0.21
39	0.19746	0.37508	0.95135	3.13941	0.18353	0.38511	0.80002	3.00918	7.59	2.60	18.9	4.33
40	0.19875	0.26707	2.13551	9.15203	0.18804	0.27064	1.61541	9.15029	5.69	1.32	32.2	0.02
41	0.19993	0.39416	1.06993	7.22941	0.18547	0.40348	0.87270	7.17014	7.80	2.31	22.6	0.83
42	0.19997	0.47670	1.05873	3.98627	0.18485	0.46229	0.87878	2.87913	8.18	3.12	20.4	38.4
43	0.19978	0.26673	2.33351	14.1542	0.18934	0.27021	1.73265	14.1535	5.51	1.29	34.6	0.00
44	0.19999	0.39847	1.07956	11.8208	0.18561	0.40637	0.88074	11.8552	7.75	1.95	22.5	0.29
45	0.20000	0.48941	1.08349	5.17858	0.18506	0.46734	0.88893	3.03688	8.07	4.72	21.8	70.5
46	0.17097	0.17097	0.61600	0.61600	0.16229	0.16229	0.53403	0.53403	5.35	5.35	15.3	15.3
47	0.19371	0.19371	1.29891	1.29891	0.18175	0.18175	0.93807	0.93807	6.58	6.58	38.4	38.4
48	0.19856	0.19856	0.72755	0.72755	0.17876	0.17876	0.54865	0.54865	11.07	11.07	32.6	32.6
49	0.19922	0.19922	1.76187	1.76187	0.18460	0.18460	1.07726	1.07726	7.92	7.92	63.5	63.5
50	0.19997	0.19997	0.77168	0.77168	0.17909	0.17909	0.55376	0.55376	11.66	11.66	39.3	39.3
51	0.19999	0.19999	0.63604	0.63604	0.17815	0.17815	0.52918	0.52918	12.26	12.26	20.1	20.1
52	0.19989	0.19989	1.87271	1.87271	0.18510	0.18510	1.09994	1.09994	7.99	7.99	70.2	70.2
53	0.20000	0.20000	0.77296	0.77296	0.17909	0.17909	0.55376	0.55376	11.67	11.67	39.5	39.5
54	0.20000	0.20000	0.63614	0.63614	0.17816	0.17816	0.52917	0.52917	12.26	12.26	20.2	20.2

Table D.7: G-limited Case: Exact and approximation

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
55	0.04545	0.04545	1.94296	1.94296	0.04545	0.04545	1.94289	1.94289	0.00	0.00	0.00	0.00
56	0.04545	0.04545	4.94296	4.94296	0.04545	0.04545	4.94289	4.94289	0.00	0.00	0.00	0.00
57	0.08333	0.08333	4.86622	4.86622	0.08333	0.08333	4.88459	4.88459	0.00	0.00	0.38	0.38
58	0.04545	0.04545	9.94296	9.94296	0.04545	0.04545	9.94289	9.94289	0.00	0.00	0.00	0.00
59	0.08333	0.08333	9.86622	9.86622	0.08333	0.08333	9.88459	9.88459	0.00	0.00	0.19	0.19
60	0.16666	0.16666	9.59375	9.59375	0.16666	0.16666	9.70580	9.70580	0.00	0.00	1.15	1.15
61	0.04545	0.04545	14.9429	14.9429	0.04545	0.04545	14.9428	14.9428	0.00	0.00	0.00	0.00
62	0.08333	0.08333	14.8662	14.8662	0.08333	0.08333	14.8845	14.8845	0.00	0.00	0.12	0.12
63	0.16666	0.16666	14.5937	14.5937	0.16666	0.16666	14.7058	14.7058	0.00	0.00	0.76	0.76
64	0.04544	0.04545	1.90796	1.94296	0.04544	0.04545	1.90792	1.94288	0.00	0.00	0.00	0.00
65	0.04545	0.04545	4.90792	4.94296	0.04545	0.04545	4.90788	4.94289	0.00	0.00	0.00	0.00
66	0.08333	0.08333	4.77300	4.86622	0.08333	0.08333	4.78675	4.88459	0.00	0.00	0.29	0.38
67	0.04545	0.04545	9.90792	9.94296	0.04545	0.04545	9.90788	9.94289	0.00	0.00	0.00	0.00
68	0.08333	0.08333	9.77298	9.86622	0.08333	0.08333	9.78672	9.88459	0.00	0.00	0.14	0.19
69	0.16665	0.16666	9.23084	9.59374	0.16665	0.16666	9.30345	9.70581	0.00	0.00	0.78	1.15
70	0.04545	0.04545	14.9079	14.9429	0.04545	0.04545	14.9078	14.9428	0.00	0.00	0.00	0.00
71	0.08333	0.08333	14.7729	14.8662	0.08333	0.08333	14.7867	14.8845	0.00	0.00	0.09	0.12
72	0.16666	0.16666	14.2305	14.5937	0.16666	0.16666	14.3031	14.7058	0.00	0.00	0.51	0.76
73	0.04515	0.04546	1.75564	1.94294	0.04515	0.04546	1.75551	1.94287	0.00	0.00	0.01	0.00
74	0.04545	0.04545	4.75025	4.94296	0.04545	0.04545	4.74961	4.94289	0.00	0.00	0.01	0.00
75	0.08298	0.08336	4.28452	4.86616	0.08299	0.08336	4.29062	4.88455	0.01	0.00	0.14	0.38
76	0.04545	0.04545	9.75024	9.94296	0.04545	0.04545	9.73717	9.94289	0.00	0.00	0.13	0.00
77	0.08333	0.08333	9.26866	9.86622	0.08332	0.08333	9.26642	9.88459	0.01	0.00	0.02	0.19
78	0.15809	0.16838	6.43116	9.58859	0.15858	0.16828	6.42519	9.70254	0.31	0.06	0.09	1.17
79	0.04545	0.04545	14.7502	14.9429	0.04545	0.04545	14.6723	14.9428	0.00	0.00	0.53	0.00
80	0.08333	0.08333	14.2685	14.8662	0.08333	0.08333	14.2188	14.8845	0.00	0.00	0.35	0.12
81	0.16351	0.16729	10.4073	14.5918	0.16378	0.16724	10.4021	14.7047	0.17	0.03	0.05	0.77
82	0.04544	0.04544	1.90796	1.90796	0.04544	0.04544	1.90793	1.90793	0.00	0.00	0.00	0.00
83	0.04545	0.04545	4.90792	4.90792	0.04545	0.04545	4.90788	4.90788	0.00	0.00	0.00	0.00
84	0.08333	0.08333	4.77300	4.77300	0.08333	0.08333	4.78674	4.78674	0.00	0.00	0.29	0.29
85	0.04545	0.04545	9.90792	9.90792	0.04545	0.04545	9.90788	9.90788	0.00	0.00	0.00	0.00
86	0.08333	0.08333	9.77298	9.77298	0.08333	0.08333	9.78673	9.78673	0.00	0.00	0.14	0.14
87	0.16665	0.16665	9.23082	9.23082	0.16665	0.16665	9.30346	9.30346	0.00	0.00	0.78	0.78
88	0.04545	0.04545	14.9079	14.9079	0.04545	0.04545	14.9078	14.9078	0.00	0.00	0.00	0.00
89	0.08333	0.08333	14.7729	14.7729	0.08333	0.08333	14.7867	14.7867	0.00	0.00	0.09	0.09
90	0.16666	0.16666	14.2305	14.2305	0.16666	0.16666	14.3031	14.3031	0.00	0.00	0.51	0.51
91	0.04515	0.04546	1.75563	1.90792	0.04515	0.04546	1.75551	1.90789	0.00	0.00	0.01	0.00
92	0.04545	0.04545	4.75025	4.90792	0.04545	0.04545	4.75002	4.90788	0.00	0.00	0.00	0.00
93	0.08298	0.08336	4.28451	4.77287	0.08299	0.08336	4.29070	4.78665	0.01	0.00	0.14	0.29
94	0.04545	0.04545	9.75024	9.90792	0.04545	0.04545	9.74808	9.90788	0.00	0.00	0.02	0.00
95	0.08333	0.08333	9.26866	9.77298	0.08333	0.08333	9.27264	9.78673	0.00	0.00	0.04	0.14
96	0.15809	0.16835	6.43111	9.21668	0.15859	0.16827	6.42525	9.29348	0.31	0.05	0.09	0.83
97	0.04545	0.04545	14.7502	14.9079	0.04545	0.04545	14.7426	14.9078	0.00	0.00	0.05	0.00
98	0.08333	0.08333	14.2685	14.7729	0.08333	0.08333	14.2416	14.7867	0.00	0.00	0.19	0.09
99	0.16351	0.16729	10.4073	14.2251	0.16379	0.16724	10.4048	14.2996	0.17	0.03	0.02	0.52
100	0.04516	0.04516	1.75554	1.75554	0.04516	0.04516	1.75542	1.75542	0.00	0.00	0.01	0.01
101	0.04545	0.04545	4.75025	4.75025	0.04545	0.04545	4.75023	4.75023	0.00	0.00	0.00	0.00
102	0.08301	0.08301	4.28394	4.28394	0.08302	0.08302	4.29030	4.29030	0.02	0.02	0.15	0.15
103	0.04545	0.04545	9.75024	9.75024	0.04545	0.04545	9.75022	9.75022	0.00	0.00	0.00	0.00
104	0.08333	0.08333	9.26865	9.26865	0.08333	0.08333	9.27517	9.27517	0.00	0.00	0.07	0.07
105	0.15863	0.15863	6.37748	6.37748	0.15961	0.15961	6.35425	6.35425	0.62	0.62	0.37	0.37
106	0.04545	0.04545	14.7502	14.7502	0.04545	0.04545	14.7502	14.7502	0.00	0.00	0.00	0.00
107	0.08333	0.08333	14.2685	14.2685	0.08333	0.08333	14.2750	14.2750	0.00	0.00	0.05	0.05
108	0.16379	0.16379	10.3589	10.3589	0.16426	0.16426	10.3542	10.3542	0.29	0.29	0.05	0.05

Table D.8: G-limited Case: Exact and approximation *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
109	0.32012	0.33832	1.44041	1.74106	0.31906	0.33913	1.42494	1.74038	0.33	0.24	1.09	0.04
110	0.33304	0.33348	4.34012	4.74405	0.33303	0.33348	4.33834	4.74408	0.00	0.00	0.04	0.00
111	0.43804	0.44942	3.79999	4.55386	0.43897	0.44875	3.86974	4.60165	0.21	0.15	1.80	1.04
112	0.33333	0.33333	9.33540	9.74423	0.33332	0.33333	9.32901	9.74423	0.00	0.00	0.07	0.00
113	0.44424	0.44460	8.67128	9.56160	0.44428	0.44457	8.75631	9.60683	0.01	0.01	0.97	0.47
114	0.54701	0.56614	7.37204	9.14736	0.54979	0.56274	7.66484	9.32738	0.51	0.60	3.82	1.93
115	0.33333	0.33333	14.3353	14.7442	0.33333	0.33333	14.2794	14.7442	0.00	0.00	0.39	0.00
116	0.44443	0.44444	13.6635	14.5618	0.44443	0.44445	13.7407	14.6070	0.00	0.00	0.56	0.31
117	0.55391	0.55760	12.0079	14.1689	0.55457	0.55678	12.3160	14.3380	0.12	0.15	2.50	1.18
118	0.27713	0.35844	1.00677	1.72096	0.26823	0.36350	0.94016	1.71648	3.32	1.39	7.09	0.26
119	0.31758	0.34120	3.05445	4.73522	0.31524	0.34237	2.90635	4.73488	0.74	0.34	5.10	0.01
120	0.36590	0.50615	2.02012	4.45976	0.35047	0.51917	1.72456	4.50288	4.40	2.51	17.1	0.96
121	0.33021	0.33489	7.19922	9.74244	0.32998	0.33500	7.10979	9.74249	0.07	0.03	1.26	0.00
122	0.38871	0.48902	3.54849	9.48385	0.37155	0.50275	2.59892	9.52464	4.62	2.73	36.5	0.43
123	0.39740	0.74949	2.49639	8.59921	0.38022	0.77364	2.19504	8.82650	4.52	3.12	13.7	2.58
124	0.33263	0.33368	11.8628	14.7438	0.33240	0.33379	11.7001	14.7437	0.07	0.03	1.39	0.00
125	0.39536	0.48371	4.59071	14.4931	0.37613	0.49909	2.97912	14.5300	5.11	3.08	54.1	0.25
126	0.39971	0.74995	2.63231	13.5505	0.38116	0.77349	2.22429	13.8041	4.87	3.04	18.3	1.84
127	0.09775	0.44278	0.19543	1.63865	0.09111	0.44652	0.17645	1.63312	7.28	0.84	10.7	0.34
128	0.09998	0.44996	0.21933	4.61407	0.09262	0.45365	0.18929	4.61303	7.95	0.81	15.8	0.02
129	0.09999	0.71309	0.20528	4.09913	0.09277	0.72060	0.19536	4.16411	7.79	1.04	5.08	1.56
130	0.10000	0.45000	0.21964	9.61379	0.09262	0.45368	0.18936	9.61288	7.96	0.81	15.9	0.01
131	0.10000	0.71989	0.20556	9.04334	0.09278	0.72570	0.19594	9.12090	7.78	0.80	4.91	0.85
132	0.10000	1.08945	0.28235	7.06257	0.09364	1.11153	0.27596	7.29536	6.79	1.99	2.31	3.19
133	0.10000	0.45000	0.21964	14.6137	0.09262	0.45368	0.18936	14.6128	7.96	0.81	15.9	0.01
134	0.10000	0.71999	0.20556	14.0416	0.09278	0.72577	0.19595	14.1197	7.78	0.80	4.90	0.55
135	0.10000	1.11406	0.28546	11.3840	0.09370	1.12769	0.27998	11.6958	6.72	1.21	1.96	2.67
136	0.27765	0.34448	1.00303	1.47649	0.26896	0.34974	0.93203	1.45870	3.23	1.50	7.62	1.22
137	0.31759	0.34096	3.05416	4.43934	0.31526	0.34223	2.90610	4.44030	0.74	0.37	5.09	0.02
138	0.36612	0.49175	2.00970	3.72582	0.35030	0.50961	1.69629	3.69244	4.52	3.51	18.4	0.90
139	0.33021	0.33489	7.19922	9.45969	0.32998	0.33500	7.10975	9.46062	0.07	0.03	1.26	0.01
140	0.38871	0.48787	3.54705	8.59397	0.37152	0.50249	2.59600	8.62431	4.63	2.91	36.6	0.35
141	0.39754	0.68177	2.39084	5.72408	0.37372	0.72740	1.96608	5.06934	6.37	6.27	21.6	12.9
142	0.33263	0.33368	11.8628	14.4642	0.33240	0.33379	11.6999	14.4641	0.07	0.03	1.39	0.00
143	0.39536	0.48360	4.59052	13.6034	0.37613	0.49908	2.97888	13.6412	5.11	3.10	54.1	0.28
144	0.39970	0.70450	2.55790	8.79050	0.37749	0.74944	2.08707	7.68376	5.88	6.00	22.5	14.4
145	0.09776	0.41179	0.19450	1.33186	0.09111	0.41351	0.17373	1.30347	7.30	0.42	11.9	2.18
146	0.09998	0.44704	0.21925	4.06215	0.09262	0.45110	0.18904	4.05474	7.95	0.90	15.9	0.18
147	0.09999	0.64238	0.20215	2.78592	0.09265	0.64623	0.18825	2.61951	7.92	0.60	7.39	6.35
148	0.10000	0.44995	0.21964	9.01345	0.09262	0.45365	0.18935	9.01973	7.96	0.82	16.0	0.07
149	0.10000	0.69098	0.20432	6.08999	0.09273	0.69906	0.19334	5.81842	7.83	1.16	5.68	4.67
150	0.10000	0.78824	0.24227	2.71653	0.09273	0.73175	0.20925	2.14763	7.83	7.72	15.7	26.4
151	0.10000	0.44999	0.21964	14.0120	0.09262	0.45368	0.18936	14.0188	7.96	0.81	15.9	0.05
152	0.10000	0.70653	0.20499	9.83031	0.09276	0.71432	0.19484	9.57719	7.80	1.09	5.21	2.64
153	0.10000	0.79765	0.24352	2.98670	0.09274	0.73449	0.20967	2.18143	7.82	8.60	16.1	36.9
154	0.09799	0.18642	0.17916	0.37507	0.09109	0.17743	0.15588	0.33695	7.57	5.07	14.9	11.3
155	0.09999	0.19944	0.19972	0.51357	0.09233	0.18749	0.16493	0.41876	8.29	6.37	21.0	22.6
156	0.09999	0.19986	0.16813	0.34682	0.09218	0.18649	0.15194	0.31443	8.48	7.17	10.6	10.3
157	0.10000	0.19999	0.19999	0.52652	0.09233	0.18772	0.16497	0.42277	8.30	6.54	21.2	24.5
158	0.10000	0.20000	0.16819	0.34838	0.09218	0.18655	0.15195	0.31490	8.48	7.20	10.6	10.6
159	0.10000	0.20000	0.16602	0.32229	0.09216	0.18638	0.15259	0.30650	8.50	7.30	8.80	5.15
160	0.10000	0.20000	0.19999	0.52666	0.09233	0.18772	0.16497	0.42278	8.30	6.54	21.2	24.5
161	0.10000	0.20000	0.16819	0.34838	0.09218	0.18655	0.15195	0.31490	8.48	7.20	10.6	10.6
162	0.10000	0.20000	0.16602	0.32229	0.09216	0.18638	0.15259	0.30650	8.50	7.30	8.80	5.15

Table D.9: G-limited Case: Exact and approximation *(continued)*

	EXACT				APPROXIMATION				ERROR (%)			
NO	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2	THR1	THR2	AB1	AB2
163	0.06059	0.06060	1.91236	1.95662	0.06059	0.06060	1.91231	1.95652	0.00	0.00	0.00	0.01
164	0.06060	0.06060	4.91232	4.95662	0.06060	0.06060	4.91227	4.95652	0.00	0.00	0.00	0.00
165	0.11110	0.11111	4.79025	4.89706	0.11110	0.11111	4.81290	4.90828	0.00	0.00	0.47	0.23
166	0.06060	0.06060	9.91232	9.95662	0.06060	0.06060	9.91226	9.95652	0.00	0.00	0.00	0.00
167	0.11111	0.11111	9.79024	9.89706	0.11111	0.11111	9.81288	9.90828	0.00	0.00	0.23	0.11
168	0.22221	0.22222	9.33365	9.67863	0.22221	0.22222	9.46608	9.74509	0.00	0.00	1.40	0.68
169	0.06060	0.06060	14.9123	14.9566	0.06060	0.06060	14.9122	14.9565	0.00	0.00	0.00	0.00
170	0.11111	0.11111	14.7902	14.8970	0.11111	0.11111	14.8127	14.9082	0.00	0.00	0.15	0.08
171	0.22222	0.22222	14.3334	14.6786	0.22222	0.22222	14.4656	14.7451	0.00	0.00	0.91	0.45
172	0.06050	0.06061	1.84247	1.95661	0.06050	0.06061	1.84240	1.95651	0.00	0.00	0.00	0.01
173	0.06060	0.06060	4.84165	4.95662	0.06060	0.06060	4.84153	4.95652	0.00	0.00	0.00	0.00
174	0.11105	0.11111	4.58874	4.89706	0.11105	0.11111	4.60420	4.90829	0.00	0.00	0.34	0.23
175	0.06060	0.06060	9.84165	9.95662	0.06060	0.06060	9.83989	9.95652	0.00	0.00	0.02	0.00
176	0.11111	0.11111	9.58770	9.89706	0.11111	0.11111	9.60137	9.90828	0.00	0.00	0.14	0.11
177	0.22120	0.22251	8.36757	9.67817	0.22132	0.22247	8.44083	9.74487	0.05	0.02	0.87	0.68
178	0.06060	0.06060	14.8416	14.9566	0.06060	0.06060	14.8378	14.9565	0.00	0.00	0.03	0.00
179	0.11111	0.11111	14.5877	14.8970	0.11111	0.11111	14.5904	14.9082	0.00	0.00	0.02	0.08
180	0.22216	0.22223	13.3208	14.6786	0.22216	0.22223	13.3924	14.7451	0.00	0.00	0.53	0.45
181	0.05369	0.06105	1.28897	1.95628	0.05355	0.06106	1.28338	1.95617	0.26	0.01	0.44	0.01
182	0.05990	0.06065	3.91530	4.95658	0.05986	0.06065	3.90516	4.95647	0.06	0.00	0.26	0.00
183	0.08778	0.11402	2.17583	4.89413	0.08721	0.11409	2.12533	4.90564	0.66	0.06	2.38	0.23
184	0.06058	0.06060	8.81914	9.95662	0.06050	0.06061	8.63024	9.95651	0.14	0.01	2.19	0.00
185	0.09542	0.11307	3.91059	9.89509	0.09499	0.11312	3.77041	9.90650	0.45	0.05	3.72	0.12
186	0.09977	0.25720	1.55423	9.62260	0.09767	0.25780	1.51903	9.69973	2.14	0.23	2.32	0.80
187	0.06060	0.06060	13.8141	14.9566	0.06050	0.06061	12.7176	14.9565	0.17	0.01	8.62	0.00
188	0.09784	0.11277	5.23252	14.8953	0.09748	0.11281	5.01584	14.9067	0.36	0.04	4.32	0.08
189	0.09998	0.25714	1.57184	14.6226	0.09786	0.25775	1.53168	14.6998	2.17	0.24	2.62	0.52
190	0.06050	0.06060	1.84247	1.92356	0.06050	0.06060	1.84240	1.92352	0.00	0.00	0.00	0.00
191	0.06060	0.06060	4.84165	4.92355	0.06060	0.06060	4.84158	4.92349	0.00	0.00	0.00	0.00
192	0.11105	0.11111	4.58874	4.80892	0.11105	0.11111	4.60422	4.81673	0.00	0.00	0.34	0.16
193	0.06060	0.06060	9.84165	9.92355	0.06060	0.06060	9.84149	9.92349	0.00	0.00	0.00	0.00
194	0.11111	0.11111	9.58770	9.80893	0.11111	0.11111	9.60262	9.81673	0.00	0.00	0.16	0.08
195	0.22120	0.22250	8.36756	9.33560	0.22132	0.22247	8.44103	9.37491	0.06	0.01	0.87	0.42
196	0.06060	0.06060	14.8416	14.9235	0.06060	0.06060	14.8412	14.9235	0.00	0.00	0.00	0.00
197	0.11111	0.11111	14.5877	14.8089	0.11111	0.11111	14.6014	14.8167	0.00	0.00	0.09	0.05
198	0.22216	0.22223	13.3208	14.3368	0.22217	0.22223	13.3945	14.3757	0.00	0.00	0.55	0.27
199	0.05369	0.06104	1.28897	1.92292	0.05355	0.06105	1.28339	1.92285	0.26	0.02	0.44	0.00
200	0.05990	0.06065	3.91530	4.92348	0.05988	0.06065	3.90917	4.92344	0.03	0.00	0.16	0.00
201	0.08778	0.11402	2.17583	4.80263	0.08722	0.11409	2.12562	4.81078	0.65	0.06	2.36	0.17
202	0.06058	0.06060	8.81914	9.92355	0.06054	0.06061	8.73096	9.92349	0.06	0.00	1.01	0.00
203	0.09542	0.11307	3.91059	9.80467	0.09495	0.11313	3.76451	9.81271	0.50	0.05	3.88	0.08
204	0.09977	0.25702	1.55427	9.18454	0.09767	0.25771	1.51896	9.23398	2.14	0.27	2.33	0.54
205	0.06060	0.06060	13.8141	14.9235	0.06056	0.06060	13.2585	14.9235	0.08	0.00	4.19	0.00
206	0.09784	0.11277	5.23252	14.8053	0.09737	0.11282	4.98587	14.8132	0.48	0.05	4.95	0.05
207	0.09998	0.25714	1.57184	14.1817	0.09786	0.25775	1.53168	14.2325	2.17	0.24	2.62	0.36
208	0.05369	0.05978	1.28878	1.65899	0.05357	0.05982	1.28298	1.65843	0.23	0.08	0.45	0.03
209	0.05990	0.06064	3.91530	4.63879	0.05989	0.06064	3.91138	4.63886	0.01	0.00	0.10	0.00
210	0.08778	0.11101	2.17519	3.83063	0.08725	0.11142	2.12233	3.82856	0.62	0.36	2.49	0.05
211	0.06058	0.06060	8.81914	9.63913	0.06058	0.06060	8.80253	9.63911	0.01	0.00	0.19	0.00
212	0.09542	0.11294	3.91054	8.68546	0.09493	0.11304	3.76165	8.69951	0.52	0.09	3.96	0.16
213	0.09976	0.18758	1.54852	3.82025	0.09749	0.18834	1.48118	3.62490	2.33	0.40	4.55	5.39
214	0.06060	0.06060	13.8141	14.6391	0.06059	0.06060	13.6925	14.6391	0.01	0.00	0.89	0.00
215	0.09784	0.11276	5.23251	13.6826	0.09733	0.11283	4.97448	13.6987	0.52	0.06	5.19	0.12
216	0.09998	0.19505	1.56719	4.80881	0.09768	0.19535	1.49629	4.41625	2.36	0.16	4.74	8.89

Table D.10: G-limited Case: Exact and approximation *(continued)*

VITA

Abdullah Daşcı was born on May 30, 1971 in Çorum, Turkey. He received his high school education at Ankara Fen Lisesi in Ankara, Turkey. He has graduated from the Department of Industrial Engineering, Bilkent University, in 1993. In October 1993, he joined to the Department of Industrial Engineering at Bilkent University as a research assistant. From that time to the present, he worked with Dr. M. Cemal Dinçer for his graduate study at the same department.